Operations Research

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Data Aggregation and Demand Prediction

Maxime C. Cohen, Renyu Zhang, Kevin Jiao

To cite this article:
Maxime C. Cohen, Renyu Zhang, Kevin Jiao (2022) Data Aggregation and Demand Prediction. Operations Research

Published online in Articles in Advance 07 Jul 2022

. https://doi.org/10.1287/opre.2022.2301

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

http://pubsonline.informs.org/journal/opre

OPERATIONS RESEARCH
Articles in Advance, pp. 1–22
ISSN 0030-364X (print), ISSN 1526-5463 (online)

Contextual Areas

Data Aggregation and Demand Prediction

Maxime C. Cohen,[a] **Renyu Zhang,**[b,*] **Kevin Jiao**[c]

[a] Desautels Faculty of Management, McGill University, Montreal, Quebec H3A 1G5, Canada; [b] Department of Decision Sciences and Managerial Economics, Business School, The Chinese University of Hong Kong, Hong Kong, China; [c] Stern School of Business, New York University, New York 10012
*Corresponding author

Contact: maxime.cohen@mcgill.ca, https://orcid.org/0000-0002-2474-3875 (MCC); philipzhang@cuhk.edu.hk, https://orcid.org/0000-0003-0284-164X (RZ); jjiao@stern.nyu.edu (KJ)

Received: June 2, 2021
Revised: December 6, 2021; February 14, 2022
Accepted: March 11, 2022
Published Online in Articles in Advance: July 7, 2022

Area of Review: Operations and Supply Chains

https://doi.org/10.1287/opre.2022.2301

Abstract. We study how retailers can use data aggregation and clustering to improve demand prediction. High accuracy in demand prediction allows retailers to effectively manage their inventory as well as mitigate stock-outs and excess supply. A typical retail setting involves predicting demand for hundreds of items simultaneously. Although some items have a large amount of historical data, others were recently introduced and, thus, transaction data can be scarce. A common approach is to cluster several items and estimate a joint model for each cluster. In this vein, one can estimate some model parameters by aggregating the data from several items and other parameters at the individual-item level. We propose a practical method referred to as *data aggregation with clustering* (DAC), which balances the tradeoff between data aggregation and model flexibility. DAC allows us to predict demand while optimally identifying the features that should be estimated at the (i) item, (ii) cluster, and (iii) aggregate levels. We show that the DAC algorithm yields a consistent and normal estimate, along with improved prediction errors relative to the decentralized benchmark, which estimates a different model for each item. Using both simulated and real data, we illustrate DAC's improvement in prediction accuracy relative to a wide range of common benchmarks. Interestingly, the DAC algorithm has theoretical and practical advantages and helps retailers uncover meaningful managerial insights.

Funding: R. Zhang is grateful for financial support from the National Natural Science Foundation of China [Grant NSFC71802133] and the Shanghai Eastern Scholar Program [Grant QD2018053].
Supplemental Material: The online appendices are available at https://doi.org/10.1287/opre.2022.2301.

Keywords: retail analytics • demand prediction • data aggregation • clustering

1. Introduction

Retailers routinely collect large volumes of historical data, which are used to improve future business practices, such as inventory management, pricing decisions, and customer segmentation. One of the most important data-driven tasks for retailers is to predict the demand for each stock-keeping unit (SKU). A common approach in practice is to classify SKUs into different departments (e.g., soft drinks) and sometimes even into subcategories (e.g., a specific types of soft drinks) and then build predictive models accordingly. A typical demand prediction model is a regression specification with the sales (or logarithmic of the sales) as the outcome variable and price, seasonality, brand, color, and promotions as features. The model coefficients are then estimated using historical data.

In many retail settings, a subset of items has been offered for a long time (referred to as "old items"), whereas other items were recently introduced. Although the demand prediction for old items is generally easy because of abundant data availability, accurately predicting the demand for newly introduced items with a limited number of historical observations is considerably more challenging. One may then wonder how the available data of old items from the same department could be leveraged to enhance the prediction of new items. Indeed, SKUs in the same department often share similar characteristics and hence tend to be affected by a particular feature in a similar way. A prominent approach is to estimate certain coefficients at an aggregate level (i.e., by gathering the data across all SKUs and assuming a uniform coefficient). For example, it seems reasonable to believe that all items in the ice cream category share the same seasonal patterns. Although this approach has been widely adopted in the retail industry, no rigorous empirical method has been developed to formalize how this data aggregation procedure should be applied for demand prediction. In this paper, we seek to bridge this gap by formalizing the tradeoff between data aggregation (i.e., pooling data from

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

different items to reduce variance) and model flexibility (i.e., estimating a different model for each item to reduce bias) in a systematic fashion.

Because of insufficient data, the traditional approach of estimating a different model for each SKU is usually inefficient for new products or SKUs with noisy observations. This approach cannot identify the right aggregation level for each coefficient and does not find the underlying cluster structure of the coefficients. Based on common clustering methods (e.g., *k*-means), we propose an efficient and integrated approach to infer the coefficient of each feature while identifying the right level of data aggregation based on the statistical properties of the estimated coefficients. Our method also allows us to incorporate multiple aggregation levels while preserving model interpretability. From a theoretical perspective, our method yields a consistent estimate, along with improved asymptotic properties. From a practical perspective, our method can easily be estimated using retail data and significantly improves out-of-sample prediction accuracy.

1.1. Main Results and Contributions

We study the tradeoff between data aggregation and model flexibility by optimally identifying the right level of aggregation for each feature, as well as the cluster structure of the items. We propose a practical method—referred to as the data aggregation with clustering (DAC) algorithm, which allows us to predict demand while optimally identifying the features that should be estimated at the (i) item, (ii) cluster, and (iii) aggregate levels. Our proposed algorithm first applies maximum-likelihood estimation to obtain a different coefficient vector for each item (called the decentralized model). It then performs a hypothesis test (i.e., t test) on the estimated coefficients from the decentralized model to identify the correct aggregation level for each feature. To characterize the cluster structure of the items, we apply the *k*-means method on the estimated coefficients from the decentralized model (as opposed to using features' average values).

We first characterize the DAC algorithm's theoretical properties. Specifically, we show that it yields a consistent estimate of the data aggregation levels and cluster structures. The estimated feature coefficients under DAC are consistent and normal. Thus, if the data have enough observations, one can correctly identify the underlying data generating process. In addition to this consistency and normality result, we derive improved prediction errors—variance, mean squared error, and generalization error are all smaller—relative to the commonly used *maximum-likelihood estimator* (MLE) method applied in a decentralized fashion to each item. Furthermore, we show that if some items have abundant data, whereas other items have limited data, our proposed algorithm improves the prediction accuracy for all items, with a more significant improvement for the items with limited data. Armed with these theoretical results, we then conduct extensive computational experiments based on both simulated and real data to illustrate the DAC algorithm's significantly improved prediction accuracy relative to 15 different benchmarks. Our results highlight the essential value of the DAC algorithm in better balancing the bias-variance tradeoff, resulting in more accurate demand prediction. Furthermore, our algorithm can accurately identify the right data aggregation levels and recover cluster structure in the nonasymptotic regime (i.e., when the sample size of the data set is finite). Finally, we apply the DAC algorithm using two years of retail data and find that it can also help retailers uncover useful insights on the relationships between the different items.

1.2. Related Literature

This paper is related to several literature streams, including prediction and clustering algorithms, retail operations, and demand forecasting.

1.2.1. Prediction and Clustering Algorithms. The problems of demand prediction and clustering have been extensively studied in the machine learning (ML) literature. Donti et al. (2017) focus on developing new ML methods by training a prediction model to solve a nominal optimization problem. Although several studies have focused on general settings, it is difficult to apply existing methods to a retail setting where multiple levels of hierarchy may exist. Elmachtoub and Grigas (2021) propose a new idea called "smart predict, then optimize" (SPO). SPO's key feature is that the loss function is computed based on comparing objective values generated by using predicted and observed data. The authors then address the computational challenge and develop a tractable SPO version. Jagabathula et al. (2018) propose a model-based embedding technique to segment a large population of customers into nonoverlapping clusters with similar preferences. Bertsimas and Kallus (2019) combine ideas from ML and operations research to propose a new prediction method. The authors solve a conditional stochastic optimization problem by incorporating various ML methods, such as local regression and random forests. Liu et al. (2021) apply clustering techniques to predict the travel time of last-mile delivery services and optimize the order assignment. Our work is also related to the traditional clustering literature. Since the introduction of *k*-means by MacQueen (1967), clustering algorithms have been extensively studied. In the context of assortment personalization, Bernstein et al. (2019) propose a dynamic clustering method to estimate customer preferences. In our paper,

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

we leverage some theoretical properties of the k-means clustering method and embed it as one of the key steps in our demand prediction algorithm.

1.2.2. Retail Operations and Demand Forecasting. Retailers are always seeking ways to improve operational decisions, such as inventory replenishment, supply chain management, and pricing. These decisions rely heavily on accurate demand prediction. Cohen et al. (2022) provide an end-to-end practical guide to leverage data analytics for demand prediction in retail. We also refer interested readers to Fildes et al. (2019b) for a comprehensive review of this literature, which demonstrates that forecasters in retailing often face the dimensionality problem of having too many features but too little data. As reported by Cohen and Lee (2020), demand uncertainty is a major issue in designing efficient global supply chains. There is an extensive body of literature focused on developing methods for demand prediction in retail settings. Sophisticated models have been developed in the last two decades to manage the increasing volume of data collected by retailers. Marketing papers, such as Van Heerde et al. (2000) and Macé and Neslin (2004), study pre- and postpromotion dips using linear regression models with lagged variables. Kök and Fisher (2007) develop a procedure to estimate substitution behavior in retail demand. Recent developments in demand prediction include the following three papers: Huang et al. (2014), who embed competitive information (including price and promotions) into demand prediction; Fildes et al. (2019a), who suggest that promotional information can be quite valuable in improving forecast accuracy; and Huang et al. (2019), who further account for the impact of marketing activities. Ma et al. (2016) develop a lasso-based four-step methodological framework to overcome the problem of the ultra-high dimensionality of the feature space under multiple product categories. In the operations management community, demand prediction models are often used as an input to an optimization problem (Cohen et al. 2017, 2021). Specifically, Cohen et al. (2017) estimate a log-log demand model using supermarket data. The authors then solve the promotion optimization problem by developing an approximation based on linear programming. It was shown in the retail operations literature that responding to accurate demand forecasts can substantially increase profits (Caro and Gallien 2010). Kesavan et al. (2010) show that incorporating the cost of goods sold, inventory, and gross margin information can substantially improve sales forecasting for retailers. In recent years, the amount of data available has grown exponentially, thus offering new opportunities for research on demand prediction (Feng and Shanthikumar 2018). In this context, our paper proposes a new demand prediction method that can efficiently aggregate data from multiple items to improve prediction accuracy.

When the demand data show too high variation or is insufficient to construct reliable forecasting models, appropriately pooling data to improve the prediction accuracy has been widely studied in the demand forecasting literature. For example, Cooper et al. (1999) and Cooper and Giuffrida (2000) propose pooled regression models and residuals to extract information and draw managerial insights on the impact of retail promotions. Dekker et al. (2004) present forecasting methods that pool demand information from a higher aggregation level and smartly combines forecasts thereof. In a time-series framework, Gür Ali et al. (2009) find the value of pooling observations from different stores to improve demand prediction accuracy. By pooling information from different stores and SKUs, Gür Ali (2013) propose a "driver moderator" that produces short-term forecasts for both existing and new SKUs. In an online fashion setting, Ferreira et al. (2016) propose nonparametric machine learning techniques to aggregate data from existing SKUs for demand prediction and price optimization of new SKUs that have never been sold before. Our main contribution in this literature is that we provide a systematic framework and an associated DAC algorithm to efficiently pool data, along with rigorous theoretical justifications and the edge of our approach over the decentralized benchmark (both theoretically and in practice), whereas most existing approaches in this literature are of heuristic nature and bear no theoretical performance guarantee or validation.

A recent stream of papers integrates a clustering step into demand prediction. For instance, Baardman et al. (2017) propose an iterative approach to cluster products and leverage existing data to predict the sales of new products. Hu et al. (2019) propose a two-step approach to first estimate the product lifecycle and then cluster and predict. Park et al. (2017) develop a generalized clusterwise linear regression model and propose algorithms to predict the demand of multiple SKUs. In our paper, however, the definition of clusters is fundamentally different. Unlike previous studies, our clustering is based on the estimated coefficients rather than the features' values. Furthermore, our model is flexible enough to account for different levels of data aggregation, whereas in previous studies, all features are essentially estimated at the cluster level. Allowing such flexibility is key to improving demand forecasting.

1.2.3. Structure of the Paper. The remainder of the paper is organized as follows. In Section 2, we introduce our model and discuss the relevant computational challenges. We then describe the DAC algorithm in Section 3. Our analytical results on the value of our proposed algorithm are presented in Section 4. In

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

Sections 5 and 6, we conduct computational experiments using simulated and real data, respectively. Our conclusions are reported in Section 7. The proofs of our analytical results are relegated to Online Appendix C.

2. Model

We introduce our demand prediction model under the generalized linear model (GLM) framework. Specifically, we consider a retail department (e.g., soft drinks or electronics) comprising n items (or SKUs). Each item has m historical observations (e.g., weekly sales transactions). We will show in Section 3 that our model and method can be straightforwardly generalized to a setting where different items have a different number of observations. For item i and observation j ($1 \le i \le n$ and $1 \le j \le m$), we denote the (log-of-)sales as $Y_{i,j}$ and the feature vector (e.g., price, promotion status, seasonality, functionality, and color) as $\boldsymbol{X}_{i,j} := (X^1_{i,j}, X^2_{i,j}, \ldots, X^d_{i,j})' \in \mathbb{R}^d$, which is independently and identically distributed (i.i.d.) with respect to observation j and independent with respect to item i. Without loss of generality, we assume that $\boldsymbol{X}_{i,j}$ has a bounded support with $\mathbb{E}[\boldsymbol{X}_{i,j}] = \boldsymbol{0} \in \mathbb{R}^d$ and a positive definite second-moment matrix $\Sigma_i := \mathbb{E}[\boldsymbol{X}_{i,j}\boldsymbol{X}'_{i,j}]$ for each item i, that is, the smallest eigenvalue $\lambda_{\min}(\Sigma_i) > 0$. This is a standard assumption to ensure identification and consistency in the statistics literature (Fahrmeir and Kaufmann 1985). The feature set is denoted by $\mathcal{D} := \{1, 2, \ldots, d\}$.

An important characteristic of our model is that a feature $l \in \mathcal{D}$ may affect the demand of an item at different data aggregation levels: (i) SKU, (ii) cluster, and (iii) department. More precisely, a feature may have the same impact on all items, captured by a uniform coefficient for all items in the department. We refer to such features as *shared* (i.e., department-level features), the set of which is denoted by $\mathcal{D}_s$. Here, we consider a setting where all the items belong to the same department to be consistent with the usual business retail practice, where demand prediction is often performed for each department separately. We highlight, however, that our approach can be directly applied to a more general setting without a department structure. Alternatively, a feature may have a different impact on different items, captured by a different coefficient for each item. We refer to such features as *nonshared* (i.e., SKU-level features), the set of which is denoted by $\mathcal{D}_n$. Finally, we assume that the items are segmented into different clusters so that some features have the same impact on items within the same cluster and a different impact on items from a different cluster. This phenomenon is captured by a uniform coefficient for all items in the same cluster (the coefficients are different for each cluster). We refer to such features as cluster-level features, the set of which is denoted by $\mathcal{D}_c$. Without loss of generality, we assume that, for each cluster-level feature l, the number of clusters k_l or the way that the clusters are formed may be different. Thus, the entire feature set, $\mathcal{D}$, can be written as the union of three disjoint sets of features that affect the demand at different aggregation levels: $\mathcal{D} = \mathcal{D}_s \cup \mathcal{D}_n \cup \mathcal{D}_c$. The aggregation structure $\mathcal{D}_s$, $\mathcal{D}_n$, and $\mathcal{D}_c$, and the cluster partition of the items are unknown a priori and will be estimated from data.

We assume that the ground truth follows the GLM specification. Specifically, the observations are generated from an exponential family distribution that includes normal, binomial, gamma, Poisson, and inverse-normal distributions as special cases. We refer to Fahrmeir and Kaufmann (1985) and McCullagh and Nelder (2019) for an introduction of the standard theory of GLM. Based on the three data aggregation levels of the features, we have

$$Y_{i,j} = G\left(\sum_{l\in\mathcal{D}_s} X^l_{i,j}\beta^s_l + \sum_{l\in\mathcal{D}_n} X^l_{i,j}\beta^n_{i,l} + \sum_{l\in\mathcal{D}_c} X^l_{i,j}\beta^c_{\varsigma(i,l),l}\right) + \epsilon_{i,j}, \quad i = 1, \ldots, n \text{ and } j = 1, \ldots, m. \tag{1}$$

Here, $\varsigma(i,l) \in \{1, 2, \ldots, k_l\}$ is the cluster index that contains item i with respect to feature l, $G(\cdot)$ represents the strictly increasing link function that establishes the relationship between the linear predictor and the mean of the outcome variable, and $\epsilon_{i,j}$ s are independent zero-mean random noises. We use $\mathcal{C}_{\varsigma,l} \subset \{1, 2, \ldots, n\}$ to denote the cluster with index ς with respect to feature l, where $\varsigma \in \{1, 2, \ldots, k_l\}$ and $\{\mathcal{C}_{1,l}, \mathcal{C}_{2,l}, \ldots, \mathcal{C}_{k_l,l}\}$ form a partition of the items $\{1, 2, \ldots, n\}$. We also call $\{\mathcal{C}_{1,l}, \mathcal{C}_{2,l}, \ldots, \mathcal{C}_{k_l,l}\}$ the cluster structure with respect to feature l. We denote $\mathcal{C}(i,l) := \mathcal{C}_{\varsigma(i,l),l}$ as the cluster that contains item i with respect to feature $l \in \mathcal{D}_c$. We also define $\mathcal{C}(i,l) := \{i\}$ if $l \in \mathcal{D}_n$, and $\mathcal{C}(i,l) := \{1, 2, \ldots, n\}$ if $l \in \mathcal{D}_s$. There are many commonly used link functions, and in practice, the function depends on the context. For example, if $Y_{i,j}$ is the number of sold units of item i in observation j, $G(u) = u$ can be the identity function and, thus, the model reduces to a linear regression. Conversely, if $Y_{i,j}$ is a binary variable, $G(u) = 1/(1 + \exp(-u))$ can be the sigmoid function. Likewise, there exist other examples of link functions, such as logarithmic and inverse squared. We assume that $\epsilon_{i,j}$ is sub-Gaussian with parameter $\sigma > 0$, that is, $\mathbb{E}[\exp(\lambda\epsilon_{i,j})] \le \exp(\lambda^2\sigma^2/2)$ for any λ, which is a standard assumption in the statistics and ML literature. We assume that all the observations are *independent* across both time periods and items. Extensions of the model and algorithm via vector autoregression (respectively, generalized least squares) to cases where observations may be correlated across time periods (respectively, items) are discussed at the end of Section 3. We also define $\beta_{i,l}$ as the coefficient of $X^l_{i,j}$ in the GLM specification in Equation (1), that is, $\beta_{i,l} = \beta^s_l$ if $l \in \mathcal{D}_s$, $\beta_{i,l} = \beta^n_{i,l}$ if $l \in \mathcal{D}_n$ and $\beta_{i,l} = \beta^c_{\varsigma(i,l),l}$ if $l \in \mathcal{D}_c$. We denote $\boldsymbol{\beta}_i := (\beta_{i,1}, \beta_{i,2}, \ldots, \beta_{i,d})'$ as the *true* coefficient vector for item i and $\boldsymbol{\beta} := (\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\beta}_n)$ as the *true* coefficient matrix for all items. Likewise, we use $\widehat{\boldsymbol{\beta}} := (\widehat{\beta}_{i,l} : 1 \le i \le n, 1 \le j \le d)$ to denote an estimator of $\boldsymbol{\beta}$. Therefore, $\widehat{\beta}_{i,l}$ is the estimator for $\beta_{i,l}$.

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

Based on the GLM specification in Equation (1), we can characterize the aggregation levels of the three types of features. For a department-level feature $l \in \mathcal{D}_s$, its coefficient β_l^s is shared among *all* items. Thus, all items in the department will have the same coefficient for this feature. In comparison, for an SKU-level feature $l \in \mathcal{D}_n$, its coefficient $\beta_{i,l}^n$ will differ across items (i.e., $\beta_{i,l}^n \neq \beta_{i,l}^n$ for $i \neq i'$). Finally, for a cluster-level feature $l \in \mathcal{D}_c$, all items in the same cluster will have the same coefficient (i.e., for all $i \neq i'$ with $\varsigma(i,l) = \varsigma(i',l) = \varsigma$, the coefficient of $X_{i,j}^l$ and that of $X_{i',j}^l$ are both $\beta_{\varsigma,l}^c$). Thus, the total number of coefficients for all n items is $d_x := n|\mathcal{D}_n| + \sum_{l \in \mathcal{D}_c} k_l + |\mathcal{D}_s| \leq nd$. We use $n_{i,l}$ to denote the number of items that share the same coefficient as item i for feature l (i.e., $n_{i,l} = 1$ if $l \in \mathcal{D}_n$, $n_{i,l} = n$ if $l \in \mathcal{D}_s$, and $n_{i,l} = |\mathcal{C}(i,l)|$ if $l \in \mathcal{D}_c$). Estimating the coefficient of some features at a certain level of aggregation is common in practice. For example, retailers often estimate seasonality coefficients at the department level to avoid overfitting and capture the fact that the items follow the same seasonal patterns. Furthermore, when estimating the effect of promotions on demand, such as cannibalization and halo effects, one may cluster several items together because promotions often have a similar impact on a group of items. Assumption 1 simplifies the exposition by avoiding the situation where two clusters have the same coefficient value. Without loss of generality, we make the following assumption throughout the paper for expositional and computational convenience.

Assumption 1. *We make the following assumptions throughout the analysis of this paper.*

(a) *If a feature l is at the SKU level (i.e., $l \in \mathcal{D}_n$), then $\beta_{i,l} \neq \beta_{i',l}$ for any pair of items i and i', that is, a SKU-level feature has a different effect on each item.*

(b) *For $l \in \mathcal{D}_c$, we have $\beta_{i,l} = \beta_{i',l}$ if and only if $\mathcal{C}(i,l) = \mathcal{C}(i',l)$, that is, a cluster-level feature has the same effect on items in the same cluster and a different effect for different clusters. Furthermore, each cluster has at least two items.*

Our main goal is to accurately predict the outcome variable Y (in our case, weekly demand) given the feature vector $\boldsymbol{X}$ (e.g., price, promotion, seasonality, or color), assuming that the data generating process follows Equation (1). Before presenting our proposed demand prediction method, we first discuss the key challenge of fitting the model by using two intuitive methods. First, one could adopt the idea of lasso regression (Tibshirani 1996, Tibshirani and Taylor 2011, Ramdas and Tibshirani 2016, Hastie et al. 2019) and estimate the coefficients through the *generalized ℓ_1−regularized MLE*. This approach revises the standard MLE by adding a generalized ℓ_1-regularizer

$$-\lambda\left(\sum_{i \neq i'} \sum_{l=1}^{d} |\beta_{i,l} - \beta_{i',l}|\right)$$

to the log-likelihood function (see Equation (19) in Online Appendix B.1). A canonical result in the statistics literature shows that ℓ_1-regularization will shrink the regularized terms to zero and thus generate sparse solutions (see, e.g., Tibshirani et al. 2005 and Zou and Hastie 2005). As a result, adding a generalized ℓ_1-regularizer to MLE may potentially be helpful to capture the fact that a feature at the aggregate or cluster level shares the same coefficient for different items. We note that this approach is in a similar spirit to the *fused lasso regression* (Tibshirani and Taylor 2011). Given the high-dimensional nature of the regularized MLE problem in Equation (19) (i.e., the number of decision variables is nd, which is at the magnitude of thousand or more in practice), estimating the coefficients is computationally prohibitive even for a linear regression specification (i.e., $G(u) = u$) as it involves inverting $(nd) \times (nd)$-matrices in each step to construct the solution path. We provide a more detailed discussion on this approach applied to our problem in Online Appendix B.1.

A second possible approach is via *direct optimization*, which is based on explicitly estimating the feature aggregation levels and cluster structures. This approach introduces one-hot encoding binary variables to represent the aggregation level of each feature and the cluster that each item belongs to. To simultaneously estimate (i) the aggregation level of each feature, (ii) the cluster that each item belongs to, and (iii) the coefficient of each feature for each item, one may either reformulate the optimization as a mixed-integer second-order conic program (SOCP), as in Equation (27) or iteratively estimate the aggregation level, cluster structure, and the coefficients by varying one of these three types of decisions and fixing the other two to maximize the maximum likelihood. In a practical setting, however, the second-order SOCP will have a too-high dimension to be computationally tractable. The iterative procedure will stop once the binary variables remain the same for two consecutive iterations. A similar iterative optimization approach was proposed by Baardman et al. (2017) to address the demand prediction problem with only cluster-level features (i.e., no department-level and no SKU-level features). In their setting, this iterative procedure was proved to converge to the true coefficients and cluster structure (i.e., the estimate is consistent). In our setting, however, the consistency of the approach proposed by Baardman et al. (2017) is not guaranteed. See Online Appendix B.2 for more details on the direct optimization approach.

3. Data Aggregation with Clustering

As mentioned, simultaneously estimating the aggregation levels, cluster structures, and feature coefficients is computationally challenging and prone to substantial prediction errors. In this section, we propose a novel approach that allows us to (a) identify the correct level of aggregation for each feature, (b) find the underlying

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

cluster structure of the SKUs with respect to each feature, and (c) generate a consistent estimate of the feature coefficients. Our method is entirely data driven and can efficiently achieve these three goals in an integrated fashion while yielding an accurate demand prediction.

We begin our analysis by focusing on a (simple) special case of the GLM in Equation (1), where all the features are at the SKU level. In this case, the data-generating process can be written as

$$Y_{i,j} = G\left(\sum_{l\in\mathcal{D}} X^l_{i,j} b_{i,l}\right) + \epsilon_{i,j},$$
$$i = 1,\dots,n \text{ and } j = 1,\dots,m. \tag{2}$$

By comparing the model specifications in (1) and (2), we have $b_{i,l} = \beta^s_l$ for $l \in \mathcal{D}_s$, $b_{i,l} = \beta^n_{i,l}$ for $l \in \mathcal{D}_n$, and $b_{i,l} = \beta^c_{\varsigma(i,l),l}$ for $l \in \mathcal{D}_c$. We refer to Model (2) as the *decentralized model* because each item is fitted in a decentralized fashion. Estimating the decentralized model is usually carried out through iterative reweighted least squares, which ultimately lead to MLE (see Online Appendix A and McCullagh and Nelder 2019 for more details). We assume that for each item, the decentralized model is well defined with a unique MLE solution, which is the case for commonly used GLMs, such as linear and logistic regression. Estimating the decentralized model can be decomposed into estimating one model for each item separately. Using the data of item i, we apply the MLE to find the estimated coefficients of this item, $\widehat{\boldsymbol{b}}_i := (\widehat{b}_{i,1}, \widehat{b}_{i,2}, \dots, \widehat{b}_{i,d})' \in \mathbb{R}^d$ as follows:

$$\widehat{\boldsymbol{b}}_i \in \arg\max_{\boldsymbol{b}_i} \sum_{j=1}^{m} \log \mathcal{L}_i(\boldsymbol{b}_i | Y_{i,j}, \boldsymbol{X}_{i,j})$$
$$= \arg\max_{\boldsymbol{b}_i} \sum_{j=1}^{m} \left[Y_{i,j} \boldsymbol{X}'_{i,j} \boldsymbol{b}_i - H(\boldsymbol{X}'_{i,j} \boldsymbol{b}_i) \right], \tag{3}$$

where $\mathcal{L}_i(\boldsymbol{b}_i | Y_{i,j}, \boldsymbol{X}_{i,j})$ is the likelihood function associated with the data $(Y_{i,j}, \boldsymbol{X}_{i,j})$ and the coefficient vector $\boldsymbol{b}_i \in \mathbb{R}^d$, and $H(\cdot) : \mathbb{R} \to \mathbb{R}$ is the infinitely differentiable normalization mapping in the GLM with $H'(u) = G(u)$ (see Online Appendix A for details). We refer to the estimator $\widehat{\boldsymbol{b}} := (\widehat{\boldsymbol{b}}_1, \widehat{\boldsymbol{b}}_2, \dots, \widehat{\boldsymbol{b}}_n)$ as the *decentralized estimator*. Throughout this paper, we parameterize the estimators with the sample size m when we want to make this dependence explicit. For example, we use $\widehat{\boldsymbol{b}}(m) := (\widehat{\boldsymbol{b}}_1(m), \widehat{\boldsymbol{b}}_2(m), \dots, \widehat{\boldsymbol{b}}_n(m))$ to denote a decentralized estimator with sample size m. Also, we define the Fisher information matrix with respect to the decentralized model of item i as

$$\mathcal{I}_i(\boldsymbol{b}_i) := -\mathbb{E}[\nabla_2 \log \mathcal{L}_i(\boldsymbol{b}_i | Y_{i,j}, \boldsymbol{X}_{i,j})],$$

where ∇_2 is the Hessian operator and the expectation is taken with respect to $(Y_{i,j}, \boldsymbol{X}_{i,j})$. We first show the following consistency and normality property of the decentralized estimator $\widehat{\boldsymbol{b}}$, which will be used as a building block for our subsequent analyses.

Proposition 1. *The decentralized estimator $\widehat{\boldsymbol{b}}$ satisfies the following properties:*

(a) *Consistency. As $m \uparrow +\infty$, (i) if $l \in \mathcal{D}_s$, then $\widehat{b}_{i,l}(m) \xrightarrow{p} \beta^s_l$; (ii) if $l \in \mathcal{D}_n$, then $\widehat{b}_{i,l}(m) \xrightarrow{p} \beta^n_{i,l}$; and (iii) if $l \in \mathcal{D}_c$, then $\widehat{b}_{i,l}(m) \xrightarrow{p} \beta^c_\varsigma(i,l),l$ where $\xrightarrow{p}$ refers to convergence in probability.*

(b) *Normality. For each item i and each feature $l \in \mathcal{D}$, there exist a threshold $\mathfrak{m}_{i,l}$ on the sample size and a constant $\psi_{i,l} > 0$ such that if $m \geq \mathfrak{m}_{i,l}$, we have, for any $\epsilon > 0$,*

$$\mathbb{P}(|\widehat{b}_{i,l}(m) - \beta_{i,l}| \leq \epsilon) \geq 1 - 3\exp(-\psi_{i,l}\epsilon^2 m). \tag{4}$$

Furthermore, for each item i, $\widehat{\boldsymbol{b}}_i$ is asymptotically normally distributed with

$$\sqrt{m}(\widehat{\boldsymbol{b}}_i(m) - \boldsymbol{\beta}_i) \xrightarrow{d} \mathcal{N}(\boldsymbol{0}, \mathcal{I}_i(\boldsymbol{\beta}_i)^{-1}) \text{ as } m \uparrow +\infty, \tag{5}$$

where $\xrightarrow{d}$ refers to convergence in distribution, $\mathcal{N}(\boldsymbol{0}, \mathcal{I}_i(\boldsymbol{\beta}_i)^{-1})$ refers to the multivariate normal distribution with mean $\boldsymbol{0} \in \mathbb{R}^d$ and covariance matrix $\mathcal{I}_i(\boldsymbol{\beta}_i)^{-1}$.

Proposition 1(a) shows that with sufficiently many observations, we can consistently estimate the feature coefficients using the decentralized model. It is unsurprising that the decentralized estimator, $\widehat{\boldsymbol{b}}$, is consistent given that the decentralized model has a high amount of flexibility (which implies a low bias; see also Fahrmeir and Kaufmann 1985). Proposition 1(b) further demonstrates that the coefficient estimation error of the decentralized approach is approximately normally distributed under both finite sample and asymptotic regimes (see Li et al. 2017 for details). Therefore, if we have sufficiently many observations for each item, the prediction accuracy of the decentralized model will be high. However, two issues remain unaddressed with the decentralized estimation: (i) how can we find the right aggregation level for each feature, and (ii) how can we identify the items' cluster structure. Furthermore, the decentralized estimation may suffer from overfitting and, hence, admit a high variance. Addressing these issues will be the main focus in the rest of this paper.

To estimate the aggregation level and the underlying cluster structure based on the GLM specification in Equation (1), we next introduce another special case of the model in which the data aggregation level and the cluster structure are known to the retailer. We refer to this case as the *aggregate model* and call its MLE the *aggregate estimator*, which we denote by

$$\widehat{\boldsymbol{b}}^a \in \arg\max_{\boldsymbol{\beta}\in\Xi} \sum_{j=1}^{m}\sum_{i=1}^{n} \log \mathcal{L}(\boldsymbol{\beta}_i | Y_{i,j}, \boldsymbol{X}_{i,j})$$
$$= \arg\max_{\boldsymbol{\beta}\in\Xi} \sum_{j=1}^{m}\sum_{i=1}^{n} \left[Y_{i,j} \boldsymbol{X}'_{i,j} \boldsymbol{\beta}_i - H(\boldsymbol{X}'_{i,j} \boldsymbol{\beta}_i) \right]$$
$$\text{where } \Xi := \{\boldsymbol{\beta} : \beta_{i,l} = \beta_{i',l} \text{ if (i) } l \in D_s \text{ or (ii) } l \in D_c$$
$$\text{and } \varsigma(i,l) = \varsigma(i',l)\}. \tag{6}$$

Thus, $\widehat{b}^a_{i,l}$ is the estimated coefficient of feature l for item i under the aggregate approach. As a counterpart of

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

Proposition 1, the following result establishes the consistency and normality of the aggregate estimator $\widehat{\boldsymbol{b}}^a$.

Proposition 2. *The aggregate estimator $\widehat{\boldsymbol{b}}^a$ satisfies the following properties:*

(a) *Consistency. As $m \uparrow +\infty$,* (i) *if $l \in \mathcal{D}_s$, then $\widehat{b}^a_{i,l}(m) \xrightarrow{p} \beta^s_l$;* (ii) *if $l \in \mathcal{D}_n$, then $\widehat{b}^a_{i,l}(m) \xrightarrow{p} \beta^n_{i,l}$; and* (iii) *if $l \in \mathcal{D}_c$, then $\widehat{b}^a_{i,l}(m) \xrightarrow{p} \beta^c_{\zeta(i,l),l}$.*

(b) *Normality. For each item i and each feature $l \in \mathcal{D}$, there exist a threshold $\tilde{\mathfrak{m}}_{i,l}$ on the sample size and a constant $\tilde{\psi}_{i,l} > 0$ such that if $m \geq \tilde{\mathfrak{m}}_{i,l}$, we have, for any $\epsilon > 0$,*

$$\mathbb{P}(|\widehat{b}^a_{i,l}(m) - \beta_{i,l}| \leq \epsilon) \geq 1 - 3\exp(-\tilde{\psi}_{i,l}\epsilon^2 m).$$

Furthermore, if $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$, then $\sqrt{m}(\widehat{\boldsymbol{b}}^a(m) - \boldsymbol{\beta})$ converges in distribution to a zero-mean degenerate multivariate normal distribution as $m \uparrow +\infty$.

Although the aggregate model requires knowing the data aggregation level of each feature and the cluster structure of the items, Proposition 1 facilitates us to infer such critical information with high accuracy from the statistical properties of the decentralized approach. We are now ready to introduce the *data aggregation with clustering* (DAC) algorithm (described in Algorithm 1), which consistently estimates the coefficient of each feature for each item, as well as correctly identifies the aggregation levels and the underlying cluster structure of the items. We denote the cumulative distribution function of a standard normal distribution by $\Phi(\cdot)$.

Algorithm 1 (Data Aggregation with Clustering $\mathsf{DAC}_{\boldsymbol{\alpha}}$)

Initialize: α = the significance level for hypothesis testing; $\widehat{\mathcal{D}}_n = \widehat{\mathcal{D}}_s = \widehat{\mathcal{D}}_c = \emptyset$.

1: (Decentralized Estimation) Estimate $\widehat{b}_{i,l}$ and its standard error $\widehat{SE}_{i,l} = \sqrt{\frac{1}{m}(\widehat{\mathcal{I}}_i(\widehat{\boldsymbol{b}}_i)^{-1})_{l,l}}$ for each item i and feature l, where $\widehat{\mathcal{I}}_i(\widehat{\boldsymbol{b}}_i)$ is the estimated Fisher information matrix for item i.

For each feature $l \in D$,

2: (Hypothesis Testing) Fix an item 1. For all other items $i \neq 1$, compute the p-value under the null hypothesis $H^l_{1,i}$ that $b_{1,l} = b_{i,l}$, which we denote as

$$p_{1,i} = 2\left[1 - \Phi\left(\frac{|\widehat{b}_{1,l} - \widehat{b}_{i,l}|}{\sqrt{\widehat{SE}^2_{1,l} + \widehat{SE}^2_{i,l}}}\right)\right].$$

Reject $H^l_{1,i}$ if and only if $p_{1,i} < \alpha$.

3: If $H^l_{1,i}$ is not rejected for all items $i \neq 1$, then assign feature l to $\widehat{\mathcal{D}}_s$.

4: If $H^l_{1,i}$ is rejected for some items $i \neq 1$ and not for others, then assign feature l to $\widehat{\mathcal{D}}_c$.

5: If $H^l_{1,i}$ is rejected for all items $i \neq 1$, then assign feature l to $\widehat{\mathcal{D}}_n$.

6: (Clustering) If $l \in \widehat{\mathcal{D}}_c$, run a one-dimensional k-means algorithm on $\{\widehat{b}_{1,l}, \widehat{b}_{2,l}, \ldots, \widehat{b}_{n,l}\}$ with $k = k_l$ and obtain the estimated cluster structure $\{\widehat{\mathcal{C}}_{1,l}, \widehat{\mathcal{C}}_{2,l}, \ldots, \widehat{\mathcal{C}}_{k_l,l}\}$.

End For.

7: (Aggregate Estimation) Obtain the aggregate estimator $\widehat{\boldsymbol{b}}^a$ based on the aggregation levels $(\widehat{\mathcal{D}}_n, \widehat{\mathcal{D}}_s, \widehat{\mathcal{D}}_c)$ and the cluster structures $\{\widehat{\mathcal{C}}_{1,l}, \widehat{\mathcal{C}}_{2,l}, \ldots, \widehat{\mathcal{C}}_{k_l,l}\}$ for each $l \in \widehat{\mathcal{D}}_c$.

Output: (a) Aggregation levels: $(\widehat{\mathcal{D}}_n, \widehat{\mathcal{D}}_s, \widehat{\mathcal{D}}_c)$, (b) Cluster structures: $\{\widehat{\mathcal{C}}_{1,l}, \widehat{\mathcal{C}}_{2,l}, \ldots, \widehat{\mathcal{C}}_{k_l,l}\}$ for each $l \in \widehat{\mathcal{D}}_c$, and (c) Feature coefficients: $\widehat{\boldsymbol{b}}^a$.

Throughout this paper, we use DAC_α to denote the data aggregation with clustering algorithm initialized with a significance level α and $\widehat{\boldsymbol{\beta}}^\alpha$ as the estimated feature coefficient matrix of the DAC_α algorithm. We also call $\widehat{\boldsymbol{\beta}}^\alpha$ the DAC_α estimator. By leveraging the consistency and normality of the decentralized estimate (see Proposition 1), we can perform hypothesis testing (i.e., the Wald test) to identify the correct data aggregation levels and cluster structure.[1] The main idea is as follows: if one cannot reject the null hypothesis that the estimated coefficients in the decentralized model $\widehat{b}_{i,l}$ and $\widehat{b}_{i',l}$ are the same, then it is very likely that either items i and i' belong to the same cluster or that feature l is an aggregate-level feature. An interesting characteristic of our method is that it uses the estimated coefficients as inputs to identify the cluster structure of the items with respect to each cluster-level feature (see Step 6), as opposed to directly using features as in traditional clustering algorithms. Identifying the cluster structure in Step 6 of Algorithm 1 is very efficient because this step runs a single-dimensional k-means. For each item i and feature $l \in \mathcal{D}$, we denote the estimated cluster that the item belongs to as $\widehat{\mathcal{C}}(i,l)$. We also highlight that the last step of the DAC algorithm to fit an aggregate model can be regularized using a lasso, ridge, or elastic net penalty to avoid unnecessary features and mitigate overfitting. This would be especially useful with high correlations between features, which is common in retail settings. Finally, we remark that the DAC algorithm is in a similar spirit as the two-stage heuristic proposed by Park et al. (2017) to address a special case of our problem, which has only cluster-level features and assumes identical cluster structures for all features. The two-stage heuristic fits the decentralized model in the first stage and then runs a multidimensional clustering algorithm for all the decentralized coefficients in the second stage. Our main contribution relative to this two-stage heuristic is multifaceted: (a) we adopt hypothesis testing to account for different data aggregation levels across features; (b) we account for different cluster structures across features; and (c) we provide rigorous theoretical justifications for the validity of our approach and the edge of our method over the decentralized benchmark.

We next show that, with an arbitrarily high probability, the DAC_α algorithm can consistently identify the aggregation level of each feature, as well as the

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

underlying cluster structure of the items with respect to each cluster-level feature. Under the DAC_α algorithm and sample size m, we define $\mathcal{E}(m,\alpha)$ as the event where the aggregation level of each feature and the cluster structure of each item for each cluster-level feature is correctly identified, namely, (i) $(\widehat{\mathcal{D}}_n, \widehat{\mathcal{D}}_s, \widehat{\mathcal{D}}_c) = (\mathcal{D}_n, \mathcal{D}_s, \mathcal{D}_c)$ and (ii) $(\widehat{\mathcal{C}}_{1,l}, \widehat{\mathcal{C}}_{2,l}, \ldots, \widehat{\mathcal{C}}_{k_l,l})$ is a permutation of $(\mathcal{C}_{1,l}, \mathcal{C}_{2,l}, \ldots, \mathcal{C}_{k_l,l})$ for each $l \in \mathcal{D}_c$. Consistent with the decentralized and aggregate approaches, the DAC_α algorithm generates a consistent and normal estimator $\widehat{\boldsymbol{\beta}}^\alpha$.

Proposition 3. *The* DAC_α *algorithm satisfies the following properties:*

(a) *Consistency. There exists a positive probability* $p(\alpha)$ *strictly decreasing in* α *with* $\lim_{\alpha \downarrow 0} p(\alpha) = 0$, *such that*

$$\lim_{m \uparrow \infty} \mathbb{P}[\mathcal{E}(m,\alpha)] \geq 1 - p(\alpha). \tag{7}$$

Furthermore, $\widehat{\boldsymbol{\beta}}^\alpha$ *is consistent, that is, as* $m \uparrow +\infty$, (i) *if* $l \in \mathcal{D}_s$, *then* $\widehat{\beta}^\alpha_{i,l}(m) \xrightarrow{p} \beta^s_l$; (ii) *if* $l \in \mathcal{D}_n$, *then* $\widehat{\beta}^\alpha_{i,l}(m) \xrightarrow{p} \beta^n_{i,l}$; *and* (iii) *if* $l \in \mathcal{D}_c$, *then* $\widehat{\beta}^\alpha_{i,l}(m) \xrightarrow{p} \beta^c_{\varsigma(i,l),l}$.

(b) *Normality. For each item* i *and each feature* $l \in \mathcal{D}$, *there exist a threshold* $\mathfrak{m}^\alpha_{i,l}$ *on the sample size and two constants* $\psi^\alpha_{i,l} > 0$ *and* $\eta^\alpha_{i,l} > 0$, *such that if* $m \geq \mathfrak{m}^\alpha_{i,l}$, *we have, for any* $\epsilon > 0$,

$$\mathbb{P}(|\widehat{\beta}^\alpha_{i,l}(m) - \beta_{i,l}| \leq \epsilon) \geq 1 - \eta^\alpha_{i,l} \exp(-\psi^\alpha_{i,l} \epsilon^2 m). \tag{8}$$

Misspecifying the feature aggregation levels for the DAC_α algorithm may stem from two types of errors in hypothesis testing (Step 2 in Algorithm 1): (i) Type I errors (i.e., rejecting the otherwise true null hypothesis) under which the algorithm falsely identifies two identical coefficients as different from each other, and (ii) Type II errors (i.e., not rejecting the otherwise false null hypothesis) under which the algorithm falsely identifies two different coefficients to be the same. By the finite-sample and asymptotic normality of the decentralized estimator (i.e., Proposition 1(b)), the Type II errors of DAC_α decay exponentially with the sample size m, which shrink to zero as m approaches infinity. The Type I errors of the DAC_α algorithm are induced by the errors in each hypothesis test, which are controlled by the significance level α. Although the Type I error probability for each $H^0_{1,i}$ is upper bounded by α, because of the notorious multiple hypothesis testing issue (Shaffer 1995), the total Type I error probability of the DAC_α algorithm is in general higher than α. Indeed, we leverage the asymptotic normality of the decentralized estimator to evaluate this probability as $p(\alpha)$, which can be arbitrarily small with a proper choice of the significance level α (see the proof of Proposition 3 for all the details). Equivalently, we can also adjust p values via Bonferroni correction from the simultaneous inference literature (Shaffer 1995) to implement the $\mathsf{DAC}_{\alpha(p)}$ algorithm. In this case, the total Type I error probability is upper bounded by p, where $\alpha(p)$ is the (strictly decreasing) inverse of $p(\alpha)$. For the practical implementation of the DAC_α algorithm using data, as we demonstrate in Sections 5 and 6, the significance level α for a single hypothesis test is a hyper-parameter to be fine-tuned via cross-validation. Finally, we remark that, for our demand prediction problem with heterogeneous data aggregation levels, Type I errors are somehow acceptable in the sense that they will only cause model imprecision (i.e., the algorithm does not identify identical coefficients), but not misspecification, so that the estimation remains consistent and normal even under these errors (Proposition 3(a) and (b)). Instead, Type I errors will only affect the efficiency of the DAC estimator, giving rise to estimates with a higher variance. We will elaborate on this point in Section 4.

Several remarks are in order regarding the DAC_α algorithm and its consistency and normality. First, Proposition 3 shows the effectiveness of Algorithm 1 under the regime where the sample size m is sufficiently large. For the case where the data sample size m is small (e.g., a new item with limited data), the estimation variance will be high. In this case, the DAC algorithm, based on hypothesis testing, may misspecify the aggregation level of each feature, giving rise to Type I errors, Type II errors, and eventually prediction errors. To address this model misspecification issue, we adopt a cross-validation procedure to fine-tune several hyper-parameters, including the significance level α for hypothesis testing and the threshold for classifying each feature into clustering and department levels (see Sections 5 and 6 for the implementation details). Using both simulated and real data, in Sections 5 and 6, we show that our proposed DAC algorithm outperforms a multitude of well-established benchmarks used in the literature and in practice. We also examine the case with new items that have limited data availability. Our theoretical results (Proposition 6) and our computational results (Figure 3) show that our DAC algorithm can efficiently pool and leverage the data while substantially improving the prediction accuracy for *all* items regardless of their data availability. More importantly, the accuracy improvement is more substantial for items with limited data.

Second, we assume in our base model that all the observations are independent across time periods and items. Such an independence assumption can also be relaxed through the vector auto-regression (VAR) model for the case with correlations across time periods and through the generalized least squares (GLS) model for the case with correlations across items. We conclude this section by discussing how the DAC algorithm can be generalized to these two cases.

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

3.1. Correlation Across Time Periods

To account for correlations across time periods, we assume that the observations $\{1,2,\ldots,m\}$ are ranked in chronological order. More precisely, the data collected in period j are indexed as observation j. We consider the following VAR model specification:

$$Y_{i,j} = \rho_i Y_{i,j-1} + \sum_{l\in\mathcal{D}_s} X^l_{i,j}\beta^s_l + \sum_{l\in\mathcal{D}_n} X^l_{i,j}\beta^n_{i,l} + \sum_{l\in\mathcal{D}_c} X^l_{i,j}\beta^c_{\varsigma(i,l)} + \epsilon_{i,j},$$
$$i = 1,\ldots,n \text{ and } j = 2,\ldots,m,$$

where $\rho_i Y_{i,j-1}$ is the auto-regression term, β^s_l, $\beta^n_{i,l}$, and $\beta^c_{\varsigma(i,l)}$ are defined in the same fashion as in the base model, and $\epsilon_{i,j}$ are the independent unobservable zero-mean sub-Gaussian error terms uncorrelated with the features $\boldsymbol{X}$ (i.e., $\mathbb{E}[\epsilon_{i,j}|\boldsymbol{X}] = 0$). Applying the standard VAR estimation approach (Greene 2003, chapter 19), we can follow the same procedure as Algorithm 1 to estimate the decentralized model, conduct hypothesis tests, cluster the items, and finally estimate the aggregate model. All the results presented in this paper remain valid under this VAR setting.

3.2. Correlation Across Items

To account for correlations across items, we consider the following GLS model specification:

$$Y_{i,j} = \sum_{l\in\mathcal{D}_s} X^l_{i,j}\beta^s_l + \sum_{l\in\mathcal{D}_n} X^l_{i,j}\beta^n_{i,l} + \sum_{l\in\mathcal{D}_c} X^l_{i,j}\beta^c_{\varsigma(i,l)} + \epsilon_{i,j},$$
$$i = 1,\ldots,n \text{ and } j = 1,\ldots,m,$$

where β^s_l, $\beta^n_{i,l}$, and $\beta^c_{\varsigma(i,l)}$ are defined in the same fashion as in the base model, and $\epsilon_{i,j}$ are the unobservable error terms with $\mathbb{E}[\epsilon_{i,j}|\boldsymbol{X}] = 0$, but $\mathbb{E}[\epsilon_{i,j}\epsilon_{i',j}|\boldsymbol{X}] \neq 0$ for $i \neq i'$. This model extension allows us to capture *complementary* and *substitute* relationships between different SKUs within a department, such as the *cannibalization effect* between dark roast and medium roast coffee and the *halo effect* between shampoo and conditioner. For this model, one can slightly modify the DAC algorithm to correctly estimate the aggregation levels, cluster structures, and feature coefficients. More specifically, we follow the same procedure as in Steps 1–6 of Algorithm 1 to estimate the decentralized model, conduct hypothesis tests, and cluster the items. However, the final step (Step 7) that estimates the aggregate model should be adjusted to apply the GLS (instead of ordinary least squares (OLS)) estimation method (Greene 2003, chapter 10). All the results presented in this paper remain valid under this GLS setting. If there are correlations across both time periods and items, we could combine the VAR and GLS frameworks and modify the DAC algorithm accordingly to yield consistent estimates of the model coefficients, data aggregation levels, and cluster structure.

4. Value of Pooling Data Through DAC

The remainder of this paper is devoted to characterizing the benefits of performing the pairwise tests and the clustering algorithm benchmarked against existing approaches in the literature. Specifically, we present three sets of analyses from different perspectives: (a) analytical comparisons between the DAC algorithm and the decentralized method, which highlight the value of data aggregation through efficient clustering; (b) simulation studies of the DAC algorithm versus several benchmarks, which show that the DAC algorithm can successfully identify and leverage data aggregation and the cluster structures; and (c) implementation of the DAC algorithm using real retail data, which showcases the practical value of our proposed method in improving demand prediction accuracy.

In this section, we examine the value of data aggregation through efficient clustering from a theoretical perspective by showing several benefits of the aggregated model relative to the decentralized model. To convey the DAC algorithm's benefits, we first observe that if the true data-generating process has aggregate and/or cluster level features, the decentralized model assumes an overly complex model and, hence, will be prone to overfitting. To formalize this intuition, we leverage the asymptotic normality property of the MLE established by Propositions 1, 2, and 3 to derive the following result on the estimation errors of the decentralized approach and the DAC algorithm.

Proposition 4. *The following statements hold:*

(a) *For the decentralized estimator* $\widehat{\boldsymbol{b}}$*, there exists a constant* $\kappa_{i,l} = (\mathcal{I}_i(\boldsymbol{\beta}_i)^{-1})_{l,l} > 0$ *for each* (i,l) *such that*

$$\lim_{m\uparrow+\infty} m\cdot\mathbb{E}(\widehat{b}_{i,l}(m) - \beta_{i,l})^2 = \kappa_{i,l},$$
$$i = 1,2,\ldots,n,\ l = 1,2,\ldots,d. \tag{9}$$

(b) *For the* DAC_α *estimator* $\widehat{\boldsymbol{\beta}}^\alpha$*, under the same set of constants* $\{\kappa_{i,l} : 1 \le i \le n, 1 \le l \le d\}$*, we have*

$$\lim_{m\uparrow+\infty} m\cdot\mathbb{E}(\widehat{\beta}^\alpha_{i,l}(m) - \beta_{i,l})^2 \le p(\alpha)\kappa_{i,l} + (1-p(\alpha))$$
$$\left(\frac{1}{n_{i,l}}\right)^2\left(\sum_{i'\in\mathcal{C}(i,l)}\kappa_{i',l}\right),\ i = 1,2,\ldots,n, l = 1,2,\ldots,d, \tag{10}$$

where $p(\alpha)$ *is the upper bound on the total Type I error of* DAC_α *characterized in Proposition 3 and the inequality is strict if* $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$*.*

(a) *If* $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$ *and* $\kappa_{i,l} > \left(\frac{1}{n_{i,l}}\right)^2\left(\sum_{i'\in\mathcal{C}(i,l)}\kappa_{i',l}\right)$*, there exists a threshold* $\bar{\mathfrak{m}}_{i,l}$ *such that if* $m \ge \bar{\mathfrak{m}}_{i,l}$*, we have*

$$\mathbb{E}(\widehat{\beta}^\alpha_{i,l}(m) - \beta_{i,l})^2 < \mathbb{E}(\widehat{b}_{i,l}(m) - \beta_{i,l})^2. \tag{11}$$

By the consistency of the decentralized and DAC estimators, when the number of observations m becomes large, the expected squared error of the estimated coefficient for each item and feature will shrink to

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

zero. The positive constant $\kappa_{i,l} = (\mathcal{I}_i(\boldsymbol{\beta}_i)^{-1})_{l,l}$ is the asymptotic variance of the decentralized estimator $\widehat{b}_{i,l}$. What makes the DAC estimator more powerful is its ability to pool the data from different items and thus further reduce the variance of the estimates, as shown by the comparison result in Proposition 4(c). We note that $\frac{1}{n_{i,l}}\sum_{i' \in \mathcal{C}(i,l)} \kappa_{i',l}$ is the average variance of the estimated coefficient of feature l for items in $\mathcal{C}(i,l)$ when using the decentralized estimator. Since $n_{i,l} \geq 2$ for $l \in D_s \cup D_c$, the condition $\kappa_{i,l} > \left(\frac{1}{n_{i,l}}\right)^2 (\sum_{i' \in \mathcal{C}(i,l)} \kappa_{i',l})$ can easily be satisfied. Furthermore, the Type I error probability is upper bounded by $p(\alpha) \approx 0$, so that the DAC algorithm yields a smaller asymptotic error relative to the decentralized estimator for the coefficients of features at the department or cluster levels. In this case, for the same training sample, the DAC algorithm will use at least twice as many observations as the decentralized estimator to estimate the coefficient of such features. Hence, the estimation error will shrink to zero faster, especially when $n_{i,l}$ is large. In practice, a typical retail department consists of a large number of items ($n > 100$), so the DAC algorithm will be much more efficient relative to the decentralized estimator for cluster- and aggregate-level features. Moreover, for these features, the value of the DAC algorithm in reducing the estimation variance of their coefficients will be stronger when the number of items n increases, because it allows to pool more data to improve the estimation efficiency of these features.

One can see from Equation (10) that the estimation error of the DAC algorithm can be decomposed into two parts. The first part, $p(\alpha)\kappa_{i,l}$, bounds the error for the case when a Type I error occurs so that the algorithm fails to identify the pooled feature coefficients. In this case, the estimation error is upper bounded by the error of the decentralized approach, $\kappa_{i,l}$, multiplied by the maximum chance that this case occurs $p(\alpha)$. The second part, $(1-p(\alpha))\left(\frac{1}{n_{i,l}}\right)^2 (\sum_{i' \in \mathcal{C}(i,l)} \kappa_{i',l})$ bounds the error for the case where the DAC algorithm correctly identifies the data aggregation levels and the cluster structures. In this case, the estimation error is upper bounded by the error of the aggregate estimator, $\left(\frac{1}{n_{i,l}}\right)^2 (\sum_{i' \in \mathcal{C}(i,l)} \kappa_{i',l})$, multiplied by the chance that the data aggregation levels and the cluster structure are correctly specified, $1-p(\alpha)$. As long as there are some aggregate level or cluster level features (i.e., $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$), then the DAC algorithm could at least partially identify some (but not all) of the pooled coefficients so that the expected error will be strictly smaller relative to the decentralized model even if a Type I error occurs. Hence, the inequality in Equation (10) is strict when $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$. Finally, we remark that the expected squared error driven Type II errors of the DAC algorithm is of a lower magnitude relative to the advantage of the aggregate estimator over the decentralized and thus can be ignored without compromising the overall error bound of the algorithm.

Proposition 4 demonstrates the significant efficiency improvement (i.e., reducing the expected estimation error) of the DAC algorithm relative to the decentralized estimator under the presence of aggregate- and cluster-level features. We next analyze how the DAC algorithm affects the mean squared error (MSE) of the predicted outcome under a fixed design (i.e., viewing the feature matrix $\boldsymbol{X}$ as *deterministic*; Rigollet 2015, chapter 2) for the linear regression model (i.e., $G(u) = u$ and $\epsilon_{i,j}$ follows an *i.i.d.* normal distribution with a mean of zero and standard deviation σ). For any estimator $\widehat{\boldsymbol{\beta}}$, we define its MSE as

$$\widehat{\mathcal{MSE}}(\widehat{\boldsymbol{\beta}}) := \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}(Y_{i,j} - \boldsymbol{X}_{i,j}\widehat{\boldsymbol{\beta}}_i)^2}{nm}.$$

Proposition 5. *Under a linear regression setting, the following statements hold:*

(a) *The MSE of the decentralized estimator $\widehat{\boldsymbol{b}}$ satisfies*

$$\mathbb{E}\left[\widehat{\mathcal{MSE}}(\widehat{\boldsymbol{b}})\right] \leq \frac{4\sigma^2 d}{m}, \tag{12}$$

where the expectation is taken with respect to the error terms $\epsilon_{i,j}$.

(b) *Assume that $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$. The MSE of the* DAC_α *estimator $\widehat{\boldsymbol{\beta}}^{\alpha}$ satisfies*

$$\mathbb{E}\left[\widehat{\mathcal{MSE}}(\widehat{\boldsymbol{\beta}}^{\alpha})\right] \leq \frac{4\sigma^2 d_x(\alpha)}{nm} + o(m^{-1}) < \frac{4\sigma^2 d}{m} \tag{13}$$

when m is sufficiently large. Here, the expectation is taken with respect to both the randomness of the error terms $\epsilon_{i,j}$ and the DAC *algorithm, $d_x(\alpha) < nd$ is the expected number of coefficients to estimate in Step 7 of Algorithm 1, which is decreasing in α and such that $\lim_{\alpha \downarrow 0} d_x(\alpha) = d_x$, and $o(m^{-1})$ is the standard "Little-o Notation."*

Comparing parts (a) and (b) of Proposition 5 reveals the insight that the expected MSE of the DAC estimator is substantially lower relative to the decentralized benchmark. Such an improvement is driven by the fact that our proposed DAC algorithm leverages hypothesis testing to pool data, hence significantly reducing the number of model coefficients to estimate (from nd to $d_x(\alpha)$), which corresponds to 20%–70%, as shown by the implementation of our algorithm using real data (Table 7). This illustrates the power of aggregating data and reducing the model dimensionality to ultimately improve prediction accuracy.

We next study an important generalization of our base model where each item i has a different number of observations m_i. This setting fits well the scenario where some items have been offered for a long time (and, hence, have abundant data), whereas other items

are new to the market (and, hence, have limited data). More specifically, we assume that, in the training set, item i has $m_i = m\tau_i$ observations for each i. Namely, the larger τ_i, the more data are available for item i. We define $\tau := \sum_{i=1}^{n} \tau_i$ and $\tau(i,l) := \sum_{i' \in \mathcal{C}(i,l)} \tau_{i'}$. Hence, $\tau(i,l) = \tau$ if $l \in \mathcal{D}_s$ and $\tau(i,l) = \tau_i$ if $l \in \mathcal{D}_n$. Then, the total number of data observations with the same coefficient for feature l as $\beta_{i,l}$ is $m(i,l) := m \cdot \tau(i,l)$. To highlight the main intuition without getting trapped in technical details, we focus on the linear regression setting with independent features. Namely, we assume that for each item i and for all data observations j, $X_{i,j}^l$ are *i.i.d.* with mean 0 and variance 1. The entire training data set is denoted as $\mathfrak{D}_{\mathtt{tr}} := \{(Y_{i,j}\, \boldsymbol{X}_{i,j}) : 1 \le i \le n, 1 \le j \le m_i\}$, whereas the training data set of item i is denoted as $\mathfrak{D}_{\mathtt{tr}}(i) := \{(Y_{i,j}\boldsymbol{X}_{i,j}) : 1 \le j \le m_i\}$.

We are interested in the *generalization error* (GE) of each item under different demand prediction methods in the random design setting, which measures the out-of-sample expected squared error assuming that the training and testing data are independently drawn from the same data generating process (Hastie et al. 2009, chapter 2.9). For an estimation algorithm $\pi \in \{\mathsf{Dec}, \mathsf{DAC}_\alpha\}$ trained on $\mathfrak{D}_{\mathtt{tr}}$, where $\mathtt{Dec}$ refers to the decentralized approach, we define the estimator generated by π as $\widehat{\boldsymbol{\beta}}(\pi, \mathfrak{D}_{\mathtt{tr}}) = (\widehat{\boldsymbol{\beta}}_1(\pi, \mathfrak{D}_{\mathtt{tr}}), \widehat{\boldsymbol{\beta}}_2(\pi, \mathfrak{D}_{\mathtt{tr}}), \ldots, \widehat{\boldsymbol{\beta}}_n(\pi, \mathfrak{D}_{\mathtt{tr}}))$, where $\widehat{\boldsymbol{\beta}}_i(\pi, \mathfrak{D}_{\mathtt{tr}})$ is the coefficient vector for item i. The GE of π for each item i is defined as follows:

$$\mathcal{GE}_i(\pi) := \mathbb{E}\left[Y_{i,m_i+1} - \boldsymbol{X}_{i,m_i+1}\widehat{\boldsymbol{\beta}}_i(\pi, \mathfrak{D}_{\mathtt{tr}})\right]^2, \quad i = 1, 2, \ldots, n,$$

where the testing data $(Y_{i,m_i+1}, \boldsymbol{X}_{i,m_i+1})$ is independently drawn from the same distribution as each training observation $(Y_{i,j}, \boldsymbol{X}_{i,j}) \in \mathfrak{D}_{\mathtt{tr}}(i)$ and the expectation is taken with respect to the randomness of both the training and testing data.

The following proposition demonstrates the effectiveness of the DAC algorithm (in the asymptotic regime) for the case where different items have different data volumes.

Proposition 6. *Under a linear regression setting, the following statements hold if the estimators are trained on* $\mathfrak{D}_{\mathtt{tr}}$*:*

(a) *The GE of the* Dec *estimator satisfies*

$$\lim_{m \uparrow +\infty} m \cdot \left(\mathcal{GE}_i(\mathsf{Dec}) - \sigma^2\right) = \frac{d \cdot \sigma^2}{\tau_i}, \qquad i = 1, 2, \ldots, n. \tag{14}$$

(b) *The GE of the* DAC_α *estimator satisfies*

$$\lim_{m \uparrow +\infty} m \cdot \left(\mathcal{GE}_i(\mathsf{DAC}_\alpha) - \sigma^2\right) \le p(\alpha) \cdot \frac{d \cdot \sigma^2}{\tau_i} + (1 - p(\alpha)) \cdot \left(\sum_{l=1}^{d} \frac{1}{\tau(i,l)}\right) \cdot \sigma^2, \; i = 1, 2, \ldots, n, \tag{15}$$

where $p(\alpha)$ *is the upper bound of the Type I error of* DAC_α *characterized in Proposition 3 and the inequality is strict if* $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$.

(c) *If* $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$*, we have, for each item* $i = 1, 2, \ldots, n$*, and sufficiently large* m*,*

$$m \cdot \left(\mathcal{GE}_i(\mathsf{Dec}) - \mathcal{GE}_i(\mathsf{DAC}_\alpha)\right) > \left(1 - p(\alpha)\right) \cdot \left(d_s\left(\frac{1}{\tau_i} - \frac{1}{\tau}\right) + \sum_{l \in \mathcal{D}_c}\left(\frac{1}{\tau_i} - \frac{1}{\tau(i,l)}\right)\right) \cdot \sigma^2 > 0. \tag{16}$$

Furthermore, $g_i(\boldsymbol{\tau}) := d_s(\frac{1}{\tau_i} - \frac{1}{\tau}) + \sum_{l \in \mathcal{D}_c}(\frac{1}{\tau_i} - \frac{1}{\tau(i,l)})$ *is convexly decreasing in* τ_i *and concavely increasing in* $\tau_{i'}$ *for each* i *and* $i' \neq i$.

Proposition 6 shows that when the sample size is sufficiently large, the DAC algorithm yields a lower generalization error relative to the decentralized approach for each item, regardless of the item's data availability. Furthermore, pooling data via our proposed DAC algorithm reduces the generalization error more substantially for items with a lower amount of data (i.e., smaller τ_i). As expected, the prediction accuracy improvement of the DAC algorithm is strengthened when the number of features at the aggregate or cluster level, d_s or d_c, is higher. We further show the robustness of this result via extensive computational experiments in Section 5.2. The monotonicity result of $g_i(\boldsymbol{\tau})$ in Proposition 6(c) also indicates that if the sample size of one item increases, other items can successfully leverage this fact but with diminishing marginal returns, whereas the item itself could extract less additional information from other items' data. The proof of Proposition 6 relies on the bias-variance decomposition of the linear regression model. We also highlight that, similar to the estimation error characterized in Proposition 4, the generalization error of the DAC algorithm can be decomposed into two parts, where the first part, $p(\alpha) \cdot \frac{d \cdot \sigma^2}{\tau_i}$, measures the GE when Type I errors occur, and the second part, $(1 - p(\alpha)) \cdot (\sum_{l=1}^{d} \frac{1}{\tau(i,l)}) \cdot \sigma^2$, measures the GE when the DAC algorithm correctly identifies aggregation levels and cluster structures. Finally, we remark that Proposition 6 benchmarks the GE of different estimators relative to σ^2, which is the variance of the noise term $\epsilon_{i,j}$ and, thus, a lower bound of any GE by relying on the bias-variance decomposition (Hastie et al. 2009, equation (7.9); Equation (54) in Online Appendix C).

To conclude this section, we note that we focused our analysis on the case where the sample size m is sufficiently large. In practice, the sample size is often limited. Ultimately, one may question whether the value of pooling data via our proposed DAC algorithm

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

remains significant in the small-sample regime. Our simulation and real data studies (Sections 5 and 6) clearly convey that the DAC algorithm efficiently identifies and leverages data aggregation and hence substantially improves the out-of-sample prediction accuracy relative to several common benchmarks.

5. Simulated Experiments

In this section, we conduct computational experiments using simulated data. We focus on the DAC algorithm's predictive power and illustrate the improvement in prediction accuracy relative to several benchmarks in the nonasymptotic regime (i.e., when the data sample size m is moderate or small). We consider two GLM settings: linear regression (i.e., $G(u) = u$; see Section 5.1) and logistic regression (i.e., $G(u) = 1/(1+\exp(-u))$; see Section 5.3). The model's performance is evaluated using the out-of-sample R^2 for linear regression and the area under the receiving operating characteristic (ROC) curve (AUC) for logistic regression. We also undertake a comprehensive sensitivity analysis to investigate how the different parameters affect the model's performance. All the results presented in this section can be reproduced by using our available code and implementation details.[2]

5.1. Linear Regression

We assume that the data are generated from the following linear model:

$$Y_{i,j} = \sum_{l\in\mathcal{D}_s} X^l_{i,j}\beta^s_l + \sum_{l\in\mathcal{D}_n} X^l_{i,j}\beta^n_{i,l} + \sum_{l\in\mathcal{D}_c} X^l_{i,j}\beta^c_{\varsigma(i,l),l} + \epsilon_{i,j},$$
$$i = 1,2,\ldots,n \text{ and } j = 1,2,\ldots,m,$$

where $\epsilon_{i,j} \sim \mathcal{N}(0,\sigma^2)$ are i.i.d. random variables. Each feature, $X^l_{i,j}$, is generated randomly from a uniform $[0,1]$ distribution, and each coefficient β is obtained from a uniform $[-5,5]$ distribution. We fix the number of clusters $k_l = 2$ for all feature $l \in \mathcal{D}$ and vary the parameters $\{n,d,m,\sigma^2,p,q\}$ one at a time (we observed similar results for alternative values of k_ls). The definition and range of values for these parameters are reported in Table 1. The parameters p and q represent the probability that a given feature is modeled at the aggregate and cluster levels, respectively (different features are drawn independently). Thus, the probability that a feature is at the SKU level is $(1-p-q)$.

Table 1. Parameters Used in Section 5.1

Parameter	Range of values
Number of items (n)	[10,150]
Number of features (d)	[5,15]
Number of observations (m)	[10,50]
Variance of the noise (σ^2)	[0.25,2]
Department-level probability (p)	[1/6,2/3] or [1/6,1/3]
Cluster-level probability (q)	[1/6,2/3] or [1/6,1/3]

It is important to note that the DAC algorithm's implementation admits three hyper-parameters (i.e., θ, R_U, and R_L) in addition to the numbers of clusters k_ls. These three parameters represent the strictness of our algorithm in determining whether a feature should be aggregated. Specifically, $\theta = \alpha$ is the p-value cutoff for statistical significance and is usually set at 0.05 or 0.01. The parameters R_U and R_L represent thresholds for the ratio of nonrejected hypotheses. For example, suppose that the percentage of nonrejected hypotheses for feature j is $R_j = 0.3$ (i.e., 30% of the items have statistically close estimated coefficients). Then, we label feature j as a department-level feature if $R_j > R_U$ and as a SKU-level feature if $R_j < R_L$. For any intermediate value $R_j \in [R_L, R_U]$, we will label feature j as a cluster-level feature. A good way to set the parameters R_L and R_U is by performing a cross-validation procedure (for more details, see Section 6). The parameters θ, R_U, and R_L provide flexibility in the tolerance level of the algorithm. In this section, we fine-tune these parameters by testing a grid of values. When implementing our algorithm with real data (Section 6), we will carefully set their values using a rigorous cross-validation procedure.

To test the performance of the DAC algorithm, we consider the following four benchmarks: decentralized, decentralized-lasso, centralized, and clustering. For each problem instance (i.e., a specific combination of $\{n,d,m,\sigma,p,q\}$), we generate 100 independent trials (i.e., data sets) and use 67% as training and 33% as testing for each trial. We then report the average out-of-sample R^2 across all items and observations. Here is a description of the methods we consider:

1. DAC: We implement our algorithm with $\theta = 0.05$, $R_U = 0.9$, and $R_L = 0.6$.
2. Decentralized: We estimate a simple OLS model for each item separately (i.e., n models).
3. Decentralized-lasso: Same as the decentralized method, but we add an ℓ_1 regularization term to each OLS model (we obtained similar results when using a regularization based on Ridge regression or Elasticnet).
4. Centralized: This is a naïve OLS model where we assume that for each feature, all the items have the same coefficient.
5. Clustering: We first cluster the items using k-means based on their features. We then fit an OLS model for each cluster.

As we can see from Figure 1, our algorithm outperforms all the benchmarks in all settings, in terms of out-of-sample R^2. Similar results were observed for the MSE. As expected, as we increase the number of observations, the prediction accuracy of the DAC algorithm will also increase. We also find that the convergence our DAC approach is faster relative to the

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

decentralized OLS method (with or without regularization), hence highlighting the benefit of the DAC algorithm under limited data availability. This clearly demonstrates the power of data aggregation and clustering present in our algorithm. Interestingly, the performance is not greatly affected by the number of items or by the number of features (indeed, the performance of each method depends on the proportion of the different feature types rather than the absolute number of features). Finally, as expected, a higher σ^2 negatively impacts the prediction accuracy of all methods, as it makes identifying the structure more challenging. Still, our proposed algorithm has a substantial advantage relative to the four benchmarks. To complement the results of Figure 1, we report some statistics on the results of the different methods. In Table 2, we report the performance statistics (measured by the out-of-sample R^2) over the 100 independent trials. In Table 3, we report the performance statistics across $n = 100$ different items (in this case, the per-item performance is captured by the MSE). As we can see from both tables, the DAC algorithm outperforms all the benchmarks for all metrics. In addition, the standard deviation—both across instances and across items—reduces significantly, hence suggesting that our proposed algorithm also decreases the variability of the performance.

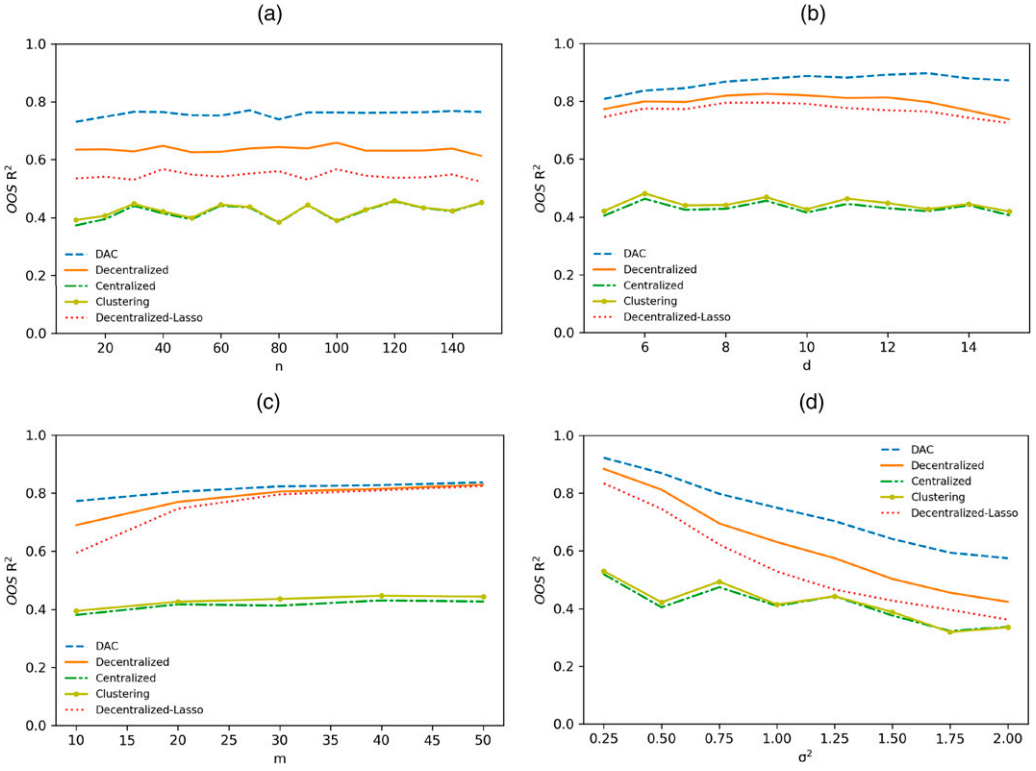
Table 2. Performance Statistics over 100 Independent Trials (Each Value Is the Out-of-Sample R^2

Method	Minimum	Maximum	Mean	Standard deviation
DAC	0.691	0.943	0.876	0.038
Decentralized	0.564	0.924	0.825	0.061
Centralized	0.025	0.907	0.403	0.199
Clustering	0.056	0.907	0.405	0.197
Decentralized-lasso	0.48	0.909	0.799	0.071

Note. Parameters: $m = 20$, $d = 8$, $n = 100$, $\sigma^2 = 1$, $p = 2/3$, $q = 1/6$.

Figure 2 presents the performance of the methods when we vary the structure probability of the features in terms of aggregation level. When a large proportion of the features are at the department level (i.e., p is getting close to one), all five methods perform well (the DAC algorithm still performs best in all cases). However, for instances where the structure is more diverse, our algorithm significantly outperforms the four benchmarks. Specifically, in all cases, the DAC algorithm leverages the structure of the problem by aggregating the data, ultimately yielding a higher prediction accuracy.

Figure 1. (Color online) Comparison of Prediction Models (for Linear Regression)



Notes. (a) Varying the number of items. (b) Varying the number of features. (c) Varying the number of observations. (d) Varying the noise magnitude.

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

Table 3. Performance Across Items (Each Value Is the Out-of-Sample MSE)

Method	Minimum	Maximum	Mean	Standard deviation
DAC	0.216	2.61	1.041	0.495
Decentralized	0.304	3.515	1.542	0.653
Centralized	0.386	13.25	3.725	3.02
Clustering	0.273	13.69	3.69	2.983
Decentralized-lasso	0.285	6.728	1.784	1.07

Note. Same parameters as Table 2.

To further evaluate the performance of the DAC algorithm, we consider two additional performance metrics: measuring the capability to correctly identify the data aggregation levels of the features, and measuring the capability to accurately recover the cluster structure of the items with respect to cluster-level features. To quantify the first capability, we investigate the proportion of features for which the DAC algorithm correctly identifies their aggregation level. This metric is defined in a similar fashion as the *Accuracy* metric used by Jagabathula et al. (2018). To quantify the second capability, we measure the ability to recover the cluster structures using the *Rand index* (Rand 1971), defined as

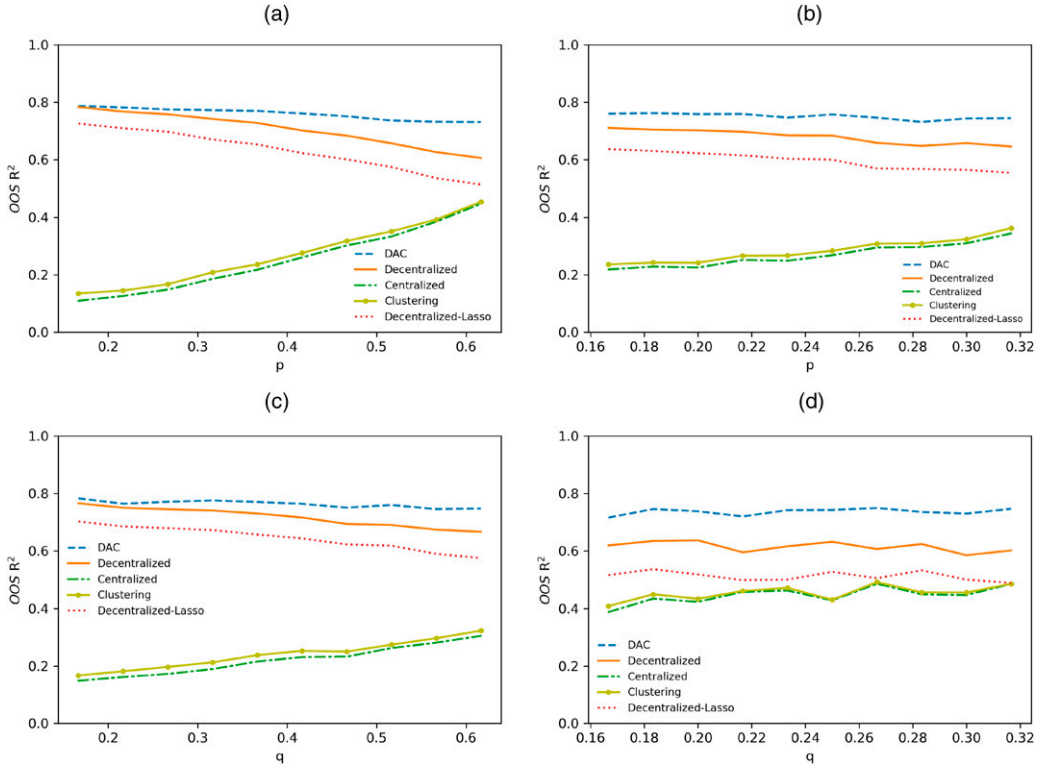
$$RI := \frac{a+b}{\binom{n}{2}},$$

where a is the number of item pairs that belong to the same cluster and predicted to be in the same cluster, b is the number of item pairs that are in different clusters and predicted to be in different clusters, and $\binom{n}{2} = n(n-1)/2$ is total number of item pairs. The higher RI is, the more capable the algorithm is in recovering the true cluster structure. As a benchmark approach to recover the cluster structure, we consider the standard k-means clustering algorithm, where we first cluster the items into different clusters based on the average normalized values of the features $\boldsymbol{X}$, and then estimate a demand model for each cluster.

We highlight that to compute the previous two metrics, one needs to have access to the ground-truth values of the aggregation levels and the cluster structure, which is not the case in settings based on real data. In such settings, the only way to evaluate the performance of our DAC algorithm is to compute the out-of-sample prediction accuracy as shown in Section 6.

We consider the same computational setting as in Table 1. We report the results of our simulation analyses in Figures 7–9 in Online Appendix D. As shown in Figure 7 in Online Appendix D, the DAC algorithm can identify the data aggregation level of each feature with a reasonably high level of accuracy. Similarly, the DAC algorithm can correctly recover the cluster structure in most instances.[3] In Figure 8 in Online

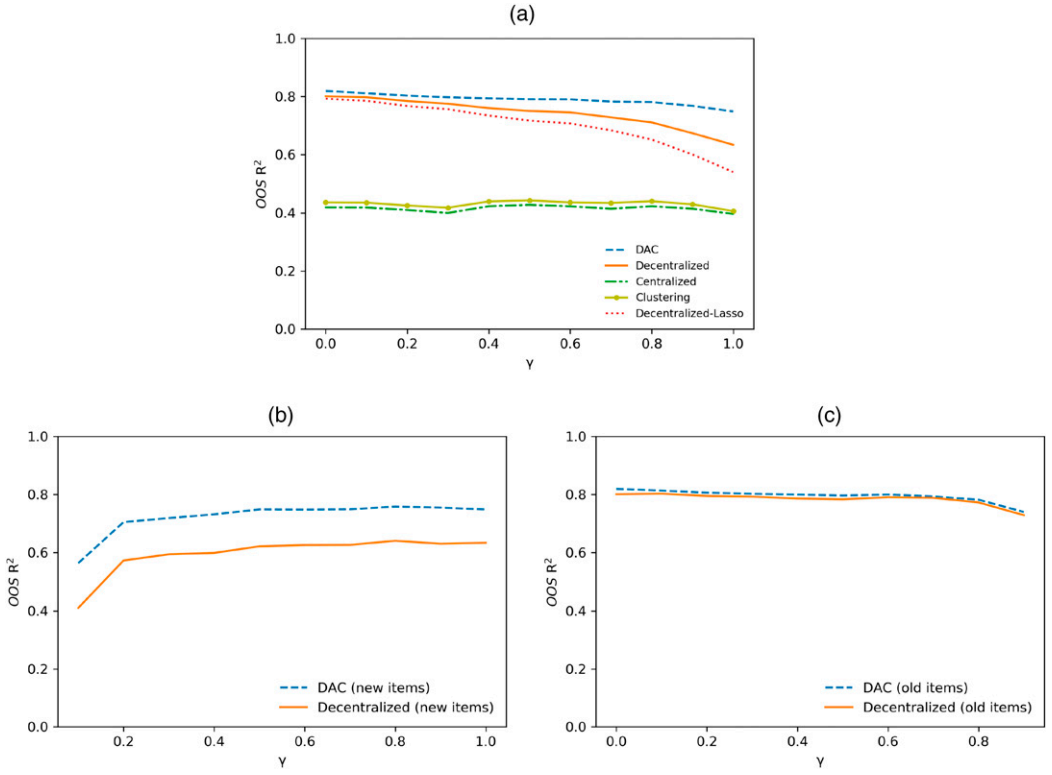
Figure 2. (Color online) Comparison of Prediction Models (for Linear Regression)



Notes. (a) Varying p (fixing $q = \frac{1}{3}$). (b) Varying p (fixing $q = \frac{2}{3}$). (c) Varying q (fixing $p = \frac{1}{3}$). (d) Varying q (fixing $p = \frac{2}{3}$).

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

Figure 3. (Color online) Comparison of Prediction Models for a Setting with Two Types of Items (for Linear Regression)



Notes. (a) All items. (b) New items only. (c) Old items only.

Appendix D, we conduct a comprehensive sensitivity analysis on both performance metrics with respect to changes in the different model parameters including the number of items n, the number of features d, the number of data observations m, the noise magnitude σ, and the number of clusters k (we assume that the number of clusters is the same for all cluster-level features). We find that the performance of our approach is robust with respect to the different parameters, generating a reasonable performance for all problem instances we examine. In particular, the DAC algorithm is most accurate in identifying aggregation levels and recovering the cluster structure when the noise magnitude is low (i.e., a small value of σ) and when the sample size m is large. This is inline with our theoretical result that the DAC algorithm is consistent in identifying aggregation levels and the cluster structure (i.e., Proposition 3(a)).

Finally, Figure 9 in Online Appendix D reports the comparison of the Rand index between the DAC algorithm and the benchmark clustering algorithm (based on k-means). Our simulation results show that the DAC algorithm outperforms the standard clustering approach in recovering the true cluster structure. Importantly, DAC dominates the benchmark with a sizable increase (15%–20%) in the Rand index.

5.2. Two Types of Items

In this section, we consider an interesting setting where a subset of the items has limited data (referred to as new items), whereas other items have abundant data (referred to as old items). Our goal is to showcase that our proposed DAC algorithm can leverage the data from the old items to improve the prediction accuracy for *both* types of items. These computational experiments complement the analytical result derived in Proposition 6 by considering a nonasymptotic regime. Specifically, we consider a setting with $n = 20$ items, $d = 5$ features, $\sigma^2 = 1$, $p = 2/3$, $q = 1/6$, $m_{new} = 10$, and $m_{old} = 30$. It thus corresponds to the situation where the old items have three times more observations than the new items. We then vary the proportion of new items, captured by the parameter $\gamma \in [0, 1]$. When $\gamma = 0$, it corresponds to the setting where all the items have abundant data (i.e., they all have $m_{old} = 30$ observations). As before, we consider 100 independent trials and use the same fine-tuned values of θ, R_U, and R_L.

The results are presented in Figure 3. In the top panel, we show the average out-of-sample R^2 across all items as a function of γ. As before, the DAC algorithm consistently outperforms all four benchmarks. As expected, the benefit of the proposed algorithm relative to the decentralized OLS method increases as γ

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

Table 4. Parameters Used in Section 5.3

Parameter	Range of values
Number of items (n)	[10,30]
Number of features (d)	[5,15]
Number of observations (m)	[40,100]

increases. In the bottom two panels, we compute the out-of-sample R^2 separately for new items (Figure 3(b)) and old items (Figure 3(c)) while focusing on the comparison between DAC and decentralized methods. The results readily confirm both insights drawn from Proposition 6: (i) the DAC algorithm improves the prediction accuracy for both types of items, and (ii) the improvement is more substantial for the items with limited data.

5.3. Logistic Regression

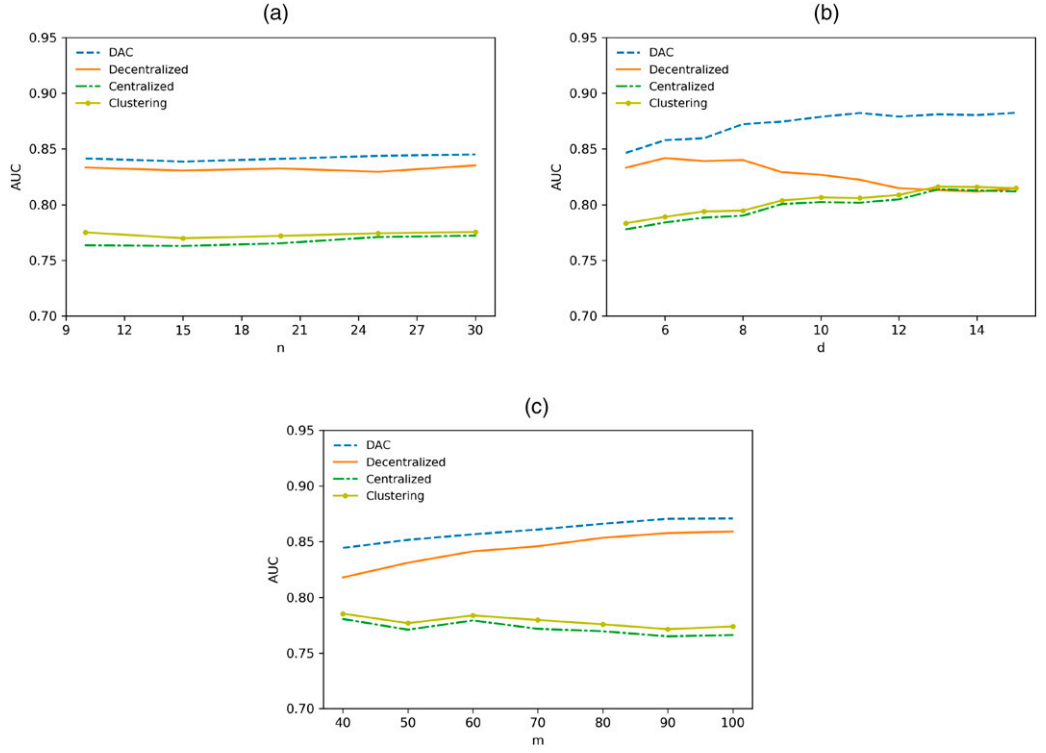
In this section, we present computational experiments for a classification problem in which the data-generating process is the logistic regression model, that is, for $i = 1,\ldots,n$ and $j = 1,\ldots,m$,

$$Y_{i,j} = \text{logit}\left(\sum_{l\in\mathcal{D}_s} X^l_{i,j}\beta^s_l + \sum_{l\in\mathcal{D}_n} X^l_{i,j}\beta^n_{i,l} + \sum_{l\in\mathcal{D}_c} X^l_{i,j}\beta^c_{\varsigma(i,l),l}\right) + \epsilon_{i,j},$$

where $\text{logit}(u) := 1/(1+\exp(-u))$ is the Sigmoid function. We consider a similar setting as in Section 5.1 and use the same values of k, θ, R_U, and R_L as before. We first generate the data matrix $\boldsymbol{X}$ from a uniform $[0,1]$ distribution and the $\boldsymbol{\beta}$ coefficients from a uniform $[-5,5]$ distribution. The outcome variable Y is then generated based on a Bernoulli distribution with parameter $\mu := \text{logit}(\boldsymbol{X\beta})$. As in the linear regression case, we systematically vary one parameter at a time. The parameters' value ranges are summarized in Table 4.

Following several prior studies on binary classification problems, we use the AUC as the metric to evaluate the performance of the different models. AUC is defined as the area under the ROC curve (Bradley 1997). It can be interpreted as the probability that a prediction model is correctly ranking a random positive outcome higher than a random negative outcome. We compare our DAC algorithm relative to three benchmarks: decentralized, centralized, and clustering (the definition of each algorithm is similar to that in Section 5.1).[4] For each instance, we generate 100 independent trials and report the average out-of-sample AUC scores. As we can see from Figure 4, our method outperforms the benchmarks in all cases. Regardless of how we vary $\{n,m,d\}$, the DAC algorithm outperforms the three other methods in terms of prediction

Figure 4. (Color online) Comparison of Prediction Models (for Logistic Regression)



Notes. (a) Varying the number of items. (b) Varying the number of features. (c) Varying the number of observations.

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

accuracy. Furthermore, a similar result as in Figure 2 was also observed for the logistic regression model (the details are omitted to avoid repetition).

To summarize, our simulation studies demonstrate a substantial and robust performance improvement for our proposed algorithm relative to several benchmarks (which are commonly used in practice and in the literature), even when the sample size is limited. Various other benchmarks will be considered in the next section, where the prediction algorithms are implemented. For both regression and classification problems, the DAC algorithm efficiently aggregates data and correctly identifies the data aggregation levels of the features and the cluster structures of the items, thus ultimately improving the prediction accuracy. In the next section, we apply the DAC algorithm using retail data to showcase its benefits in a practical setting.

6. Applying the DAC Algorithm to Retail Data

In this section, we apply the DAC algorithm to a retail data set from a large global retailer (we cannot reveal the name of the retailer because of a nondisclosure agreement). We first provide a detailed description of the data and then test the prediction performance of the DAC algorithm relative to a broad range of benchmarks (we consider a total of 15 commonly used benchmarks). Finally, based on our computational findings, we draw managerial insights that can help retailers infer which features should be aggregated in practice.

6.1. Data

We have access to the retailer's online sales data. The data set includes the weekly sales information of three departments between November 2013 and October 2016. A typical department comprises 100–150 SKUs. In addition to the weekly sales information, the data set includes the weekly price, a promotion indicator (i.e., whether an item was promoted), the vendor, and the color of the SKU. Table 5 summarizes the specifics of each department. The size corresponds to the number of items in each department, and the numbers in parentheses are the standard deviations.

As we can see from Table 5, each department has a large number of observations (here, the observations are at the SKU-week level). There is also a great variation in terms of weekly sales, prices, promotion frequency, and discount rates across the different departments. Table 6 provides a brief description of the different fields in our data set. The effective weekly price is computed as the total weekly revenue divided by the total weekly sales. Functionality is a segmentation hierarchy used by the firm to classify several SKUs from the same department into subcategories.

Table 6. Fields in Our Data Set (Observations Are Aggregated at the SKU-Week Level)

Fields	Description
SKU ID	Unique SKU ID
Week	Week index
Year	Year index
Units	Total weekly sales of a specific SKU
Price	Effective weekly price of a specific SKU
PromoFlag	Whether there was a promotion during that week
Functionality	Class index of a specific SKU
Color	Color of a specific SKU
Vendor	Vendor of a specific SKU

Based on the features available in our data, we consider the following model specification:

$$\begin{aligned} Y_{i,t} = {} & \beta^i_{\text{Trend}} \cdot T_{i,t} + \beta^i_0 \cdot p_{i,t} + \beta^i_1 \cdot \text{PromoFlag}_{i,t} \\ & + \beta^i_2 \cdot \text{Fatigue}_{i,t} + \beta^i_3 \cdot \text{Seasonality}_{i,t} + \\ & + \beta^i_4 \cdot \text{Functionality}_{i,t} + \beta^i_5 \cdot \text{Color}_{i,t} \\ & + \beta^i_6 \cdot \text{Vendor}_{i,t} + \epsilon_{i,t}. \end{aligned} \tag{17}$$

Equation (17) includes the following features: $Y_{i,t}$ is the total weekly sales of item i in week t (our dependent variable), $T_{i,t}$ is the trend variable of item i (we normalize the year so that $T_i = 0, 1, 2, 3$), $p_{i,t}$ is the effective price of item i in week t, $\text{PromoFlag}_{i,t}$ is a binary variable indicating whether there is a promotion for item i in week t, $\text{Fatigue}_{i,t}$ is the number of weeks since the most recent promotion for item i (if there is no previous promotion, $\text{Fatigue}_{i,t} = 0$; this feature allows us to capture the postpromotion dip effect), Seasonality is a categorical variable that measures the weekly or monthly effect on sales (we use one-hot encoding), and $\epsilon_{i,t}$ is an additive unobserved error. The remaining three variables are categorical variables indicating the web class index (functionality), color, and vendor of the SKU, respectively.

Table 5. Summary Statistics of the Data from Each Department

Department	Size	Observations	Weekly sales	Price	Promotion frequency	Discount rate
1	147	19,064	108.11 (377.45)	42.92 (38.80)	31.3%	4.2%
2	134	20,826	254.23 (517.07)	8.94 (8.67)	7.7%	6.4%
3	125	14,457	68.68 (691.55)	97.61 (67.12)	8.5%	1.5%

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

6.2. Prediction Performance

6.2.1. Benchmarks. As in Section 5, we compare the performance of the DAC algorithm relative to the same four benchmarks (decentralized, decentralized-lasso, centralized, and clustering). In this section, for our analysis with real data, we also consider the following additional benchmarks: decentralized-elasticnet, DBSCAN clustering, OPTICS clustering, centralized-ISI (item-specific intercepts), centralized- K -means CFE (cluster -fixed effects), centralized-DBSCAN CFE, centralized-OPTICS CFE, decision tree, random forest, gradient boosted tree, and neural network.[5] We consider three different clustering methods as benchmarks (k-means, DBSCAN, and OPTICS) and ML-based methods. Finally, we test adding fixed effects to the decentralized model under several configurations. We thus compare the DAC algorithm's performance to a total of 15 benchmarks, which are commonly used in both academia and practice.

It is important to mention that when implementing our DAC algorithm, we need to slightly adapt the clustering step of the algorithm. To ensure that we output a single cluster structure, we first collect the estimated coefficients from all cluster-level features and then fit a multidimensional k-means model (instead of a one-dimensional k-means for each feature). To avoid over-fitting, one can also include a ℓ_1-regularization term in the aggregated estimation (Step 7 of Algorithm 1).

6.2.2. Implementation of DAC. Because the DAC algorithm has four hyper-parameters (k, θ, R_U, R_L), we use an extensive cross-validation procedure to fine-tune these parameters and select the best model. For each department, we first randomly split the data into training (70%) and testing (30%) sets. We assume that each design parameter lies within a prespecified range: $k \in \{3, 4, \ldots, 10\}$, $\theta \in \{0.01, 0.05, 0.1, 0.5\}$, $R_U \in \{0.7, 0.8, 0.9\}$, and $R_L \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ (we add the option $\theta = 0.5$ to include the decentralized method as a special case). For each combination of hyper-parameters, we perform a five-fold cross-validation by fitting the model on 80% of the training data and compute the R^2 based on the remaining 20% of the data. This procedure is repeated five times for each parameter combination, and we compute the average R^2 over the five folds. We next select the best model based on the average cross-validation performance. Finally, the out-of-sample R^2 is computed on the testing set. Furthermore, because the train/cross-validation/test split is done randomly, we conduct 100 independent trials and report the mean and the 95% confidence interval of the out-of-sample R^2.

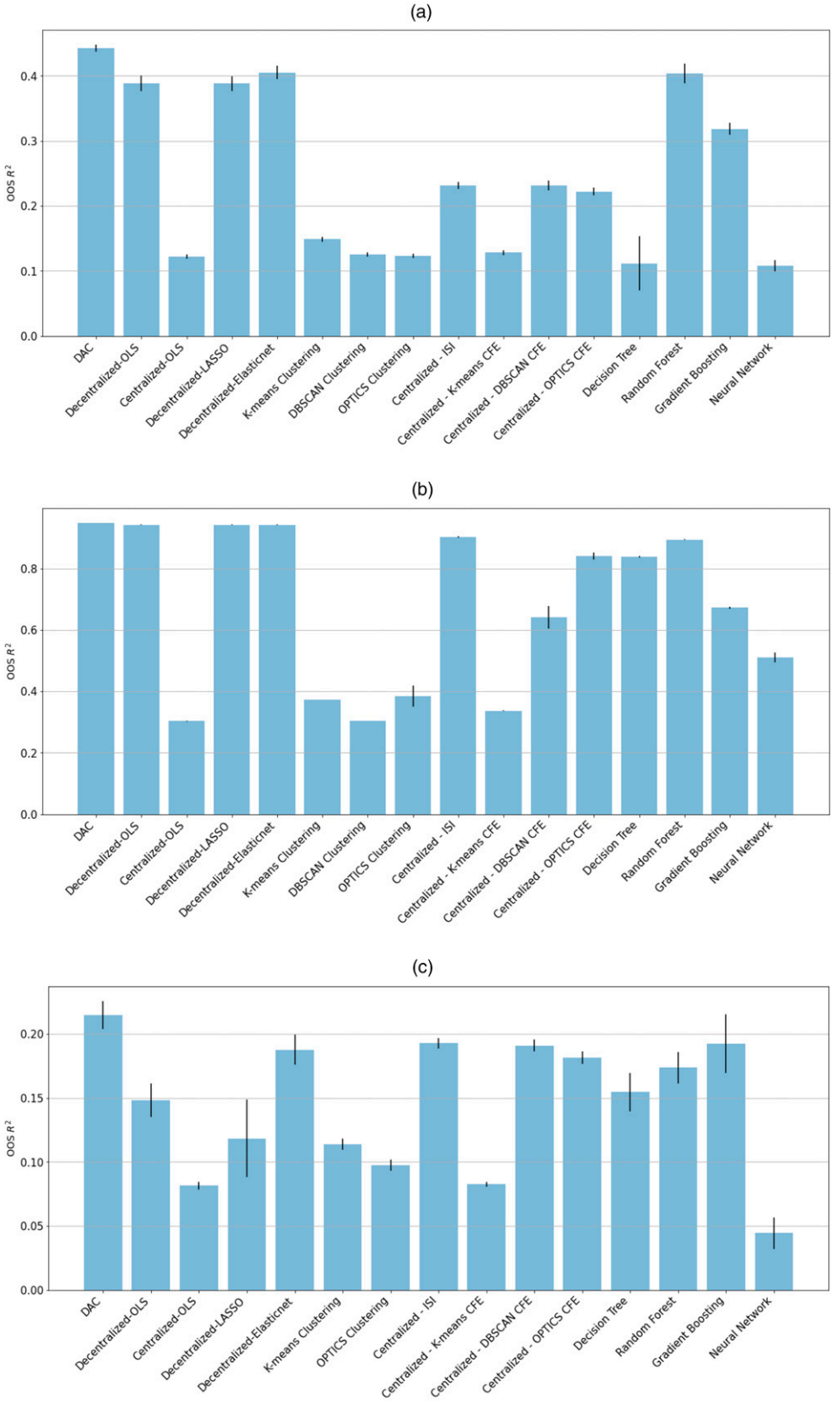
Our computational environment relies on the resources of Compute Canada and, more precisely, on the Volta generation Nvidia V100-SXM2-16Go node accessible on the Beluga computing network, with 6 Go of memory (V100 offers the performance of up to 100 CPUs in a single GPU).

6.2.3. Results. We present the results for the out-of-sample R^2 using a five-fold cross validation in Figure 5. As expected, we obtain similar results for the MSE and the mean absolute percentage error (the details are omitted for conciseness). In Figure 5, each bar represents the average out-of-sample performance, and the length of the vertical line corresponds to the 95% confidence interval across 100 trials. For all three departments, the DAC algorithm not only achieves a better average prediction performance but also often has a smaller variance. This shows that our algorithm is robust to different train/test splits, which is very desirable in practice. In addition, DAC outperforms all 15 benchmarks regardless of data quality. More precisely, Department 2 seems to have high-quality data and predictability power, whereas the data for Department 3 seems to be of lower quality (the number of observations per SKU is the smallest for Department 3, and data variability is high). Irrespective of the data quality, the DAC algorithm yields a clear improvement in prediction accuracy relative to all the benchmarks we considered. For completeness, we also recorded the in-sample R^2 values and consistently observed that the DAC algorithm reduces the amount of overfitting relative to the other methods.

6.2.4. Time-Based Split Results. Thus far, we considered testing the different methods by using a random-based split. This allows us to perform a cross validation and ultimately compute confidence intervals to make statistical claims on the performance comparisons. An alternative way is to use a time-based split. Specifically, we can split the data by using the first Y% weeks for training (e.g., $Y = 70$) and the remaining $1 - Y$% for testing. This alternative data splitting rule is attractive in terms of prediction applicability but lacks statistical support in terms of comparing the different methods. When opting for a time-based split, two options are possible: an item-based split or an absolute split. In an item-based split, we divide the data into training and testing sets for each item separately. In an absolute split, we look at the total number of weeks in the data set and then divide the data for all the items using the same week threshold. These two options differ when different items are introduced to the assortment at different times. In Figure 6, we present the results for Department 2 (the highest performing department) under an item-based split using 70% for training and 30% for testing. We obtained similar results for other split ratios and when using an absolute split. When using a time-based split, we can also consider time-series methods. Specifically, we include the Prophet

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

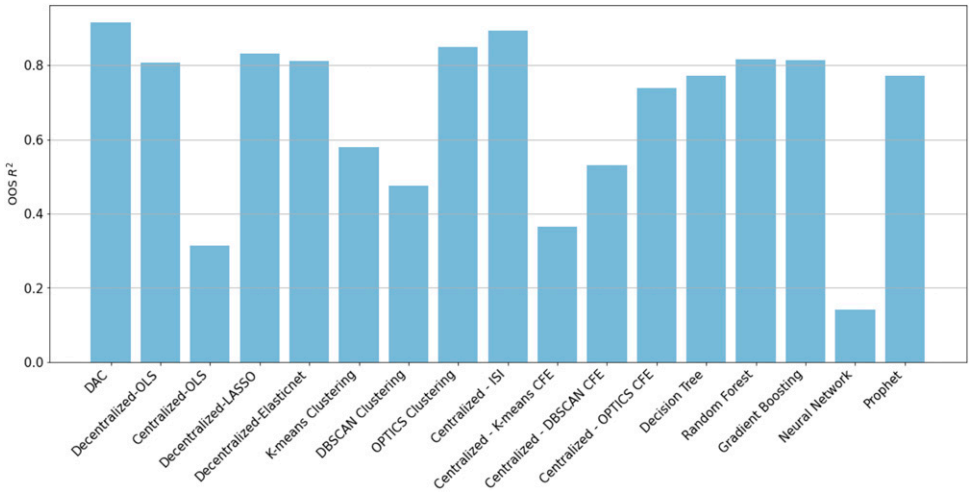
Figure 5. (Color online) Performance Comparison Using Real Data (Metric: Out-of-Sample R^2, Random Split)



Notes. (a) Department 1. (b) Department 2. (c) Department 3.

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

Figure 6. (Color online) Comparison Using Real Data for Department 2 (Metric: Out-of-Sample R^2, Time-Based Split)



method as an additional benchmark (Taylor and Letham 2018), which is considered among the state-of-the-art time-series approaches for demand prediction. Overall, as in the random-based split, the DAC algorithm outperforms all the benchmarks.

In summary, our results based on real data from three retail departments strongly suggest that the DAC algorithm has superior performance in terms of prediction accuracy. In addition to outperforming 15 benchmarks, our method reduces overfitting and provides a great interpretability advantage. Specifically, it can help retailers identify the correct level of data aggregation for the different features and uncover the underlying cluster structure. Unlike other methods, the model output yields meaningful managerial insights, as we discuss next.

6.3. Managerial Insights

Thus far, we focused on the prediction performance of our proposed method. We then apply the DAC algorithm to our data set and examine the estimation output. Our goal is to draw managerial insights into the hierarchical structure of the features. Next, we summarize our findings.

- The DAC algorithm can significantly reduce the model dimension. In Table 7, we report the number of estimated coefficients for the decentralized and DAC approaches across all three departments. Depending on the department, the number of estimated coefficients is reduced by 20%–70%. The results in Table 7 confirm that shared coefficients do occur in practice and that data aggregation can play an important role in correctly identifying the aggregation structure.
- Practitioners often argue that seasonality features should be aggregated at the department level for demand prediction (Cohen et al. 2017, Vakhutinsky et al. 2019). Using our retail data set, we discover that this is indeed the case. If we model seasonality at the month level (i.e., we use 11 dummy variables for each calendar month), we find that for two departments, all 11 variables should indeed be estimated at the department level (whereas for the third department, the same is true for 7 of 11 variables). If we instead model seasonality at the week level (i.e., we use 51 dummy variables for each week of the year), we find that 48 of 51 variables should be estimated at the department level for two departments (and 18 of 51 for the third department). Thus, our findings validate and refine a well-known insight in practice.
- The price feature is unarguably one of the most important features for demand prediction in retail. According to our estimation results, for all three departments, we obtain a distinct coefficient for the price feature, implying that the price coefficient should be estimated at the SKU level. This typically holds for departments with a heterogeneous item collection.
- We find that the fatigue and promotion features should be estimated at the department level for all three departments (except the fatigue feature for one department). This is an interesting insight that can guide retailers when deciding their promotion strategy.
- We also find that all vendor and color dummy variables should be estimated at the SKU level. This is unsurprising given that most vendor-color combinations are unique for a specific SKU.
- The functionality dummy variables have different aggregation levels and cluster structures. Interestingly,

Table 7. Number of Estimated Coefficients

Department	Decentralized	DAC
1	9,408	6,095
2	6,298	4,977
3	3,000	892

Published in Operations Research on July 07, 2022 as DOI: 10.1287/opre.2022.2301.
This article has not been copyedited or formatted. The final version may differ from this version.

most cluster-level features come from functionality. One possible explanation is that the functionality feature is obtained based on the hierarchy structure used by the company. Thus, SKUs with similar characteristics are usually labeled under the same functionality, making the cluster structure more prominent for functionality features. Retailers can use such results to potentially revise and improve their hierarchical structure and product segmentation.

7. Conclusion

Demand prediction or sales forecasting is an important task faced by most retailers. Improving prediction accuracy and drawing insights on data aggregation can significantly impact retailers' decisions and profits. When designing and estimating predictive models, retailers need to decide the aggregation level of each feature (e.g., seasonality and price). Some features may be estimated at the SKU level, others at the department level, and the rest at a cluster level. Traditionally, this problem was addressed by trial-and-error or by relying on past experience. It is common to see data scientists testing a multitude of model specifications until they find the best aggregation level for each feature. Such an ad hoc approach can be tedious and is not scalable for cases with a large number of features and items. The goal of this paper is to develop an efficient method to simultaneously determine (i) the correct aggregation level of each feature, (ii) the underlying cluster structure, and (iii) the estimated coefficients.

We propose a method referred to as the (DAC) algorithm. The DAC algorithm can determine the correct aggregation level and identify the cluster structure of the items. This method is tractable even when data dimensionality is high, and it can significantly improve the efficiency in estimating the model parameters. We first derive several analytical results to demonstrate the validity and benefits of our proposed method. Specifically, we show that the DAC algorithm yields a consistent estimate, along with improved asymptotic properties relative to the decentralized method. We then go beyond the theory and implement the DAC algorithm using a large retail data set. In all our computational tests, we observe that the proposed method significantly improves prediction accuracy relative to a multitude of benchmarks. Finally, we convey that our method can help retailers uncover useful insights from their data.

Acknowledgments

The authors thank Paul-Emile Gras and Arthur Pentecoste who helped us conduct the computations presented in Sections 5 and 6; the retail partner for sharing data; and Compute Canada for allowing us to use their computing resources.

Endnotes

[1] Under Assumption 1(b), each cluster has at least two items. Thus, Step 4 of Algorithm 1 (with a time complexity of $O(nd)$) is sufficient to determine the SKU-level features. If we relax this assumption (i.e., some cluster may have only one feature), the DAC algorithm can easily be adapted to run $O(n^2d)$ pairwise hypothesis tests instead of $O(nd)$.

[2] See https://github.com/DACPublicator/DAC_Publication.

[3] We note that the notion of "good performance" in this context is subjective, as it depends on the performance of alternative methods. We will consider below a benchmark approach for the task of recovering the cluster structure and convey the superiority of our approach.

[4] Because estimating a decentralized model with ℓ_1 regularization is computationally prohibitive for the logistic regression setting, we only show the performance of the decentralized model without regularization.

[5] The implementation details of the aforementioned methods are quite generic and are available upon request.

References

Baardman L, Levin I, Perakis G, Singhvi D (2017) Leveraging comparables for new product sales forecasting. Preprint, submitted September 30, https://dx.doi.org/10.2139/ssrn.3086237.

Bernstein F, Modaresi S, Sauré D (2019) A dynamic clustering approach to data-driven assortment personalization. *Management Sci.* 65(5):2095–2115.

Bertsimas D, Kallus N (2019) From predictive to prescriptive analytics. *Management Sci.* 66(3):1025–1044.

Bradley AP (1997) The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition* 30(7):1145–1159.

Caro F, Gallien J (2010) Inventory management of a fast-fashion retail network. *Oper. Res.* 58(2):257–273.

Cohen MA, Lee HL (2020) Designing the right global supply chain network. *Manufacturing Service Oper. Management* 22(1):15–24.

Cohen MC, Kalas JJ, Perakis G (2021) Promotion optimization for multiple items in supermarkets. *Management Sci.* 67(4):2340–2364.

Cohen MC, Gras PE, Pentecoste A, Zhang R (2022) *Demand Prediction in Retail: A Practical Guide to Leverage Data and Predictive Analytics* (Springer, Berlin).

Cohen MC, Leung NHZ, Panchamgam K, Perakis G, Smith A (2017) The impact of linear optimization on promotion planning. *Oper. Res.* 65(2):446–468.

Cooper LG, Giuffrida G (2000) Turning datamining into a management science tool: New algorithms and empirical results. *Management Sci.* 46(2):249–264.

Cooper LG, Baron P, Levy W, Swisher M, Gogos P (1999) PromocastTM: A new forecasting method for promotion planning. *Marketing Sci.* 18(3):301–316.

Dekker M, Van Donselaar K, Ouwehand P (2004) How to use aggregation and combined forecasting to improve seasonal demand forecasts. *Internat. J. Production Econom.* 90(2):151–167.

Donti P, Amos B, Kolter JZ (2017) Task-based end-to-end model learning in stochastic optimization. *Adv. Neural Inform. Processing Systems* 30:5484–5494.

Elmachtoub AN, Grigas P (2021) Smart "predict, then optimize." *Management Sci.* 68(1):9–26.

Fahrmeir L, Kaufmann H (1985) Consistency and asymptotic normality of the maximum likelihood estimator in generalized linear models. *Ann. Statist.* 13(1):342–368.

Feng Q, Shanthikumar JG (2018) How research in production and operations management may evolve in the era of big data. *Production Oper. Management* 27(9):1670–1684.

Ferreira KJ, Lee BHA, Simchi-Levi D (2016) Analytics for an online retailer: Demand forecasting and price optimization. *Manufacturing Service Oper. Management* 18(1):69–88.

Fildes R, Goodwin P, Önkal D (2019a) Use and misuse of information in supply chain forecasting of promotion effects. *Internat. J. Forecasting* 35(1):144–156.

Fildes R, Ma S, Kolassa S (2019b) Retail forecasting: Research and practice. *Internat. J. Forecasting*.

Greene WH (2003) *Econometric Analysis* (Pearson Education India).

Gür Ali Ö (2013) Driver moderator method for retail sales prediction. *Internat. J. Inform. Tech. Decision Making* 12(06):1261–1286.

Gür Ali Ö, Say S, Van Woensel T, Fransoo J (2009) SKU demand forecasting in the presence of promotions. *Expert Systems Appl.* 36(10):12340–12348.

Hastie T, Tibshirani R, Wainwright M (2019) *Statistical Learning with Sparsity: The Lasso and Generalizations* (CRC Press, Boca Raton, FL).

Hastie T, Tibshirani R, Friedman JH, Friedman JH (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, vol. 2 (Springer, Berlin).

Hu K, Acimovic J, Erize F, Thomas DJ, Van Mieghem JA (2019) Forecasting new product life cycle curves: Practical approach and empirical analysis. *Manufacturing Service Oper. Management* 21(1):66–85.

Huang T, Fildes R, Soopramanien D (2014) The value of competitive information in forecasting FMCG retail product sales and the variable selection problem. *Eur. J. Oper. Res.* 237(2):738–748.

Huang T, Fildes R, Soopramanien D (2019) Forecasting retailer product sales in the presence of structural change. *Eur. J. Oper. Res.* 279(2):459–470.

Jagabathula S, Subramanian L, Venkataraman A (2018) A model-based embedding technique for segmenting customers. *Oper. Res.* 66(5):1247–1267.

Kesavan S, Gaur V, Raman A (2010) Do inventory and gross margin data improve sales forecasts for us public retailers? *Management Sci.* 56(9):1519–1533.

Kök AG, Fisher ML (2007) Demand estimation and assortment optimization under substitution: Methodology and application. *Oper. Res.* 55(6):1001–1021.

Li L, Lu Y, Zhou D (2017) Provably optimal algorithms for generalized linear contextual bandits. *Internat. Conf.on Machine Learn.*, 2071–2080.

Liu S, He L, Max Shen ZJ (2021) On-time last-mile delivery: Order assignment with travel-time predictors. *Management Sci.* 67(7):4095–4119.

Ma S, Fildes R, Huang T (2016) Demand forecasting with high dimensional data: The case of SKU retail sales forecasting with intra- and inter-category promotional information. *Eur. J. Oper. Res.* 249(1):245–257.

Macé S, Neslin SA (2004) The determinants of pre- and postpromotion dips in sales of frequently purchased goods. *J. Marketing Res.* 41(3):339–350.

MacQueen J (1967) Some methods for classification and analysis of multivariate observations. *Proc. 5th Berkeley Sympos. on Math. Statist. and Probability*, vol. 1, 281–297.

McCullagh P, Nelder JA (2019) *Generalized Linear Models* (Routledge, Oxfordshire, England).

Park YW, Jiang Y, Klabjan D, Williams L (2017) Algorithms for generalized clusterwise linear regression. *INFORMS J. Comput.* 29(2):301–317.

Ramdas A, Tibshirani RJ (2016) Fast and flexible ADMM algorithms for trend filtering. *J. Comput. Graphical Statist.* 25(3): 839–858.

Rand WM (1971) Objective criteria for the evaluation of clustering methods. *J. Amer. Statist. Assoc.* 66(336):846–850.

Rigollet P (2015) High dimensional statistics lecture notes. Unpublished notes.

Shaffer JP (1995) Multiple hypothesis testing. *Annu. Rev. Psych.* 46(1):561–584.

Taylor SJ, Letham B (2018) Forecasting at scale. *Amer. Statist.* 72(1): 37–45.

Tibshirani R (1996) Regression shrinkage and selection via the lasso. *J. Royal Statist. Soc. B* 58(1):267–288.

Tibshirani RJ, Taylor J (2011) The solution path of the generalized lasso. *Ann. Statist.* 39(3):1335–1371.

Tibshirani R, Saunders M, Rosset S, Zhu J, Knight K (2005) Sparsity and smoothness via the fused lasso. *J. Royal Statist. Soc. Ser. B Statist. Methodology* 67(1):91–108.

Vakhutinsky A, Mihic K, Wu SM (2019) A prescriptive analytics approach to markdown pricing for an e-commerce retailer. *J. Pattern Recognition Res.* 14(1):1–20.

Van Heerde HJ, Leeflang PS, Wittink DR (2000) The estimation of pre- and postpromotion dips with store-level scanner data. *J. Marketing Res.* 37(3):383–395.

Zou H, Hastie T (2005) Regularization and variable selection via the elastic net. *J. Royal Statist. Soc. Ser. B Statist. Methodology* 67(2): 301–320.

Maxime C. Cohen is the Scale AI chair professor of retail and operations management and codirector of the Retail Innovation Laboratory at the Desautels Faculty of Management at McGill University. His core expertise lies at the intersection of data science and operations. He has collaborated with Google, Waze, Oracle Retail, IBM Research, Via, Spotify, Aldo Group, Circle K, and Staples and serves on the advisory boards of several startups.

Renyu Zhang is an associate professor at The Department of Decision Sciences and Managerial Economics, The Chinese University of Hong Kong. His recent research develops data-driven optimization, causal inference, and machine learning methodologies to evaluate and optimize the operations strategies of online platforms.

Kevin Jiao received his PhD from New York University, The Stern School of Business. His doctoral research was on data-driven models with applications in crowdsourcing, retail, and peer-to-peer lending.